UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Análisis de Regresión y Algoritmos Genéticos

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

MATEMÁTICO

PRESENTA:

MAX BENJAMIN AUSTRIA SALAZAR

TUTOR

M. EN C. ANTONIO SORIANO FLORES

CIUDAD UNIVERSITARIA, CD. MX., 2021

1. Datos del alumno:

Apellido paterno	Austria
Apellido materno	Salazar
Nombre(s)	Max Benjamín
Teléfono	57004278
Universidad	Universidad Nacional Autónoma de México
Facultad	Facultad de Ciencias
Carrera	Matemáticas
Número de Cuenta	308028342

2. Datos del tutor:

Grado	M. en C.
Nombre(s)	Antonio
Apellido paterno	Soriano
Apellido materno	Flores

3. Datos del sinodal 1:

Grado	MTIA.
Nombre(s)	Alejandro
Apellido paterno	Martínez
Apellido materno	López

4. Datos del sinodal 2:

Grado	M. en C.
Nombre(s)	Ana Paulina
Apellido paterno	Pérez
Apellido materno	Romero

5. Datos del sinodal 3:

Grado	Fís.
Nombre(s)	Jimmy
Apellido paterno	Hernández
Apellido materno	Morales

6. Datos del sinodal 4:

Grado	Act.
Nombre(s)	Yolanda
Apellido paterno	Martínez
Apellido materno	Guerrero

7. Datos del trabajo escrito:

Titulo	Análisis de Regresión y Algoritmos Genéticos.
Subtitulo	-
Número de páginas	110
Año	2021

"El Tao que puede ser dicho
no es el Tao verdadero"
Lao Tse

Agradecimientos

> “ El agradecimiento es la memoria del corazón”
>
> Lao Tse

No me gusta tener que escribir los agradecimientos, pues de acuerdo a mi primera cita, las palabras no pueden representar mis verdaderos sentimientos o intenciones hacia las personas que voy a mencionar. Pero solo por no dejar pasar la oportunidad, escribiré para ustedes.

Me gustaría agradecer a todas las personas que hicieron esto posible que terminara mis estudios de licenciatura, pero me limitare a unas cuantas:

- Agradezco a la UNAM por el aprendizaje.
- A mi asesor, Antonio, por su paciencia y comprensión.
- A mis sinodales por sus aportes.
- A mis amigos Adriana, Brianda, Dulce, Ruth y Víctor, realmente me divertí con ustedes y significan mucho para mí.
- A Miriam, eres tan fuerte y tan compresiva al mismo tiempo. Has sido un ejemplo.
- A Yolanda, eres muy noble, espero que realmente encuentres lo que quieres.
- A mis hermanos (Diana e Ismael), les deseo lo mejor (sé que pueden conseguirlo).
- A mis sobrinos (Dael y Daniel), se merecen todo lo bueno de la vida.
- A mis padres (Elvia y Javier), les agradezco el esfuerzo que han hecho por mí y en respuesta solo puedo decir voy a hacer que valga la pena.

¡Gracias!

Índice general

Índice de cuadros

Índice de figuras

Introducción

Los algoritmos genéticos son técnicas de optimización inspiradas en la teoría de la evolución, que forman parte de las llamadas metaheurísticas, y han resultado muy útiles en problemas de optimización combinatoria. Estos procedimientos operan a través de búsquedas iterativas y estocásticas, imitando (con operadores) cada uno de los agentes que producen evolución, como lo son: mutaciones, deriva genética, migración y selección natural. Resultan una opción atractiva por su capacidad de aproximar soluciones de problemas "NP" [1] en un tiempo razonable.

Un algoritmo genético puede ser clasificado de acuerdo a su función objetivo en dos clases: mono-objetivo, cuando la función a optimizar tiene por codominio a los números reales, y multiobjetivo, sí su codominio es un espacio vectorial real. En el segundo caso, un concepto muy relevante es el de dominancia de Pareto, pues a través de este se define el criterio de optimalidad, sin hacer uso de un orden total de los elementos (pues $\mathbb{R}^n$ no es un conjunto ordenable).

En este trabajo se explora la relación entre los algoritmos genéticos multiobjetivo y el análisis de regresión lineal multiple, principalmente en la selección de modelos y en la regresión dispersa no convexa. Pues son problemas de optimización combinatoria donde se busca un subconjunto de variables para crear un modelo que sea adecuado de acuerdo a sus respectivas funciones objetivo. La selección de modelos se guía primordialmente de encontrar modelos que minimicen criterios de selección y la regresión dispersa no convexa penaliza la suma de cuadrados sumando número de variables explicativas multiplicado por un factor positivo.

El objetivo de esta tesis es mostrar y justificar el uso de un algoritmo genético multiobjetivo en el proceso de selección de modelo de regresión lineal múltiple. Para alcanzar este propósito, se argumentará que es válido el uso del algoritmo demostrando que la Frontera de Pareto que calcula (para la función objetivo que se propondrá) contiene todos los "buenos modelos" (para la definición de un "buen modelo" que se plantea). Donde, ejemplos de buenos modelos son: el que seleccionaría el AIC, el BIC, el Cp de Mallows ó modelos de regularización dispersa no convexa, entre otros.

La exposición de los temas se divide como a continuación se muestra:

Análisis de Regresión: En el primer capítulo se expone la teoría de análisis de regresión lineal múltiple (sólo resultados conocidos).

- De la Sección 1 a la 5 se muestran algunos elementos básicos como: los estimadores,

[1] El lector interesado en esta clase de problemas puede dirigirse a Garey, M., Johnson., D. (1979).

el teorema de Gauss-Márkov y pruebas de hipótesis asociadas a los supuestos del modelo lineal.

- En la Sección 6 se presentan los tres modelos lineales con regularización, dos de ellos muy conocidos (Lasso,Regresión dispersa no convexa y Ridge), y la relación que tienen con el estimador tradicional.

Algoritmos Genéticos: El capítulo de algoritmos genéticos muestra los conceptos necesarios para comprender el funcionamiento de ésta clase de procedimientos de optimización.

- De las Secciones 7 a la 11 se expone la teoría general de algoritmos genéticos.
- En la 12 se indica los mecanismos del algoritmo de optimización multiobjetivo NSGA-II con el propósito de hacer comprensibles las modificaciones que se realizan en la Sección 13.
- Particularmente, la Subsección 13.2 presenta dos resultados originales: se especifica la función objetivo que se utiliza en la optimización multiobjetivo y se muestra su relación con la regresión dispersa no convexa .

Selección de Modelos: En esta parte de la tesis se presentan los algoritmos y criterios clásicos de selección de modelo, pero tambien se exponen elementos originales del trabajo como:

- Definición de "Buen criterio".
- Ejemplos de "buenos criterios" como AIC, BIC y Cp de Mallows.
- Y una proporción que vincula los "buenos criterios" a la frontera de Pareto.

Simulaciones: El capitulo "Simulaciones" se escribió para respaldar los resultados alcanzados en el presente trabajo; para esto se realizaron 100 simulaciones donde se genera de manera aleatoria datos que permiten emular el proceso de selección de modelo de regresión lineal múltiple y se analiza la información generada.

Conclusiones: Finalmente se presentan los resultados de las simulaciones del NSGA-II adaptado a regresión lineal, algunas observaciones sobre el uso del algoritmo en la práctica y las conclusiones finales.

Análisis de regresión

En este capítulo se parte de una explicación general de los objetivos que se tienen en el proceso de modelado de datos, se listan (a manera de ejemplo) algunas técnicas de utilidad, se presenta de manera breve los resultados más básicos del análisis de regresión lineal múltiple: los supuestos del modelo, los estimadores, el teorema de Gauss-Márkov, pruebas de hipótesis y por último se exponen modelos lineales con regularización (particularmente la regresión dispersa no convexa).

1. El problema de regresión

En investigación cuantitativa muchas veces se desea conocer el proceso generador del fenómeno de estudio (o de una parte bien definida del mismo) para poder describirlo, analizarlo, predecirlo o comprender la relación que existe entre las variables del problema de investigación. La mayoría de las veces este proceso es desconocido o inobservable, pero investigarlo metodológicamente es posible, primero es necesario que las observaciones estén estructuradas en variables $\mathbf{X}_1, \ldots, \mathbf{X}_p$ para poder plantear el problema en términos matemáticos.

Así, la cuestión consiste en encontrar una relación desconocida $R \subseteq \prod_{i=1}^{p} \mathbf{X}_i$, que caracterice el comportamiento conjunto de las variables de acuerdo con el objetivo de investigación. Si en el análisis, una variable destaca como dependiente principal ($\mathbf{Y}$), se puede plantear a la relación (R) como una función y a la variable dependiente ($\mathbf{Y}$) como imagen de R.

Para dar una función adecuada, se debe cumplir que:

- El dominio y codomino sean apropiados para las variables, es decir, que cubra todos los posibles valores que estas puedan tomar.
- El algoritmo de búsqueda de R debe ser lo suficientemente robusto para que sea capaz de encontrar una función sobre el espacio que planteen las restricciones del problema, y además:
- Contar con un criterio para se pueda cumplir que:
    - R ajusta suficientemente bien a los datos.
    - Se evita el sobre ajuste, que sucede cuando R describe únicamente los datos contenidos en las observaciones pero no al proceso.

En resumen:

Sea $\mathbf{Y}$ una variable aleatoria que se le nombrará variable dependiente o respuesta y sean $\mathbf{X}_1, \ldots, \mathbf{X}_p$ variables a las que se les llamará variables predictivas, covariables o regresoras. Se abreviará $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_p)$ al vector de variables predictivas. El problema de regresión consiste en aproximar la función R tal que $Y = R(\mathbf{X})$, es decir, estimar

$$R(\mathbf{x}) := \mathbb{E}[\mathbf{Y}|\mathbf{X} = \mathbf{x}] = \int yp(y|\mathbf{x})d\mathbf{x}.$$

2. Regresión lineal múltiple

El análisis de regresión lineal múltiple es una de las técnicas mas renombradas dentro del análisis de datos (en la teoría y la práctica), básicamente se supone que la variable respuesta ($\mathbf{Y}$) es una combinación lineal de las variables regresoras $\mathbf{X}_1, \ldots, \mathbf{X}_p$ y se desea encontrar el vector de parámetros que determinen de manera única a ésta función lineal.

Entre las principales ventajas del análisis de regresión lineal se encuentran:

- Puede predecir.
- Reconoce las variables relevantes de las que no aportan.
- Modela de forma relativamente sencilla.
- Y el uso práctico de la misma cuenta con una amplia guía teórica.

Y su principal desventaja es:

- No todos los fenómenos son lineales o linealizables por lo que no se puede aplicar a cualquier tipo de datos.

Algunas otras técnicas usadas en regresión o extracción de patrones (descritas de manera muy burda) son:

Redes neuronales Una técnica inspirada en el funcionamiento del sistema nervioso, donde de igual forma se tienen covariables $\mathbf{X}_i$ para una variable respuesta $\mathbf{Y}$ (ambas en $\mathbb{R}$). En ésta técnica no se supone linealidad o una forma funcional entre la variable respuesta y las variables explicativas. Para funciones continuas cuenta con algunos resultados de análisis matemático que garantizan la existencia de una red (de un tipo específico) que sea tan próxima a la variable dependiente ($\mathbf{Y}$) como se quiera (si se tienen las variables correctas), para la clasificación cuenta con otros resultados (véase Barron, A. R., [1989]). Ésta técnica ha tenido éxito en problemas donde las técnicas más clásicas resultan difíciles de aplicar como clasificación o reconocimiento de patrones en imágenes, voz ó procesamiento de señales, y son una opción a otros como el pronóstico de series económicas o financieras. El proceso para obtener los parámetros de una red es lento (en comparación con el análisis de regresión lineal) y se debe tener consideraciones para evitar el sobre-ajuste, el modelo no cuenta con interpretación (únicamente capacidad predictiva).

Regresión simbólica Es una aplicación de la programación genética al problema de regresión. En ésta, dados símbolos y operaciones, se busca una relación mediante una ecuación implícita $f(\mathbf{Y}, \mathbf{X}_1, \ldots, \mathbf{X}_n) = 0$ de la variable respuesta $\mathbf{Y}$ y sus covariables $\mathbf{X}_i$, esto tanto en la forma de la ecuación como en los parámetros. Se puede tomar Icke, I., Bongard, J. C., (2013) como texto de referencia.

Modelos de series temporales Cuando la variable dependiente ($\mathbf{Y}$) tenga una estructura dinámica (que dependa del tiempo) se han propuesto múltiples modelos estadísticos para modelarla ($\mathbf{Y}_t$), como los ARIMA, GARCH, VAR, o los modelos de espacios de estados. Cada uno con sus propios supuestos y resultados para la selección de modelo, algunos de ellos aceptan covariables $\mathbf{X}_i$ continuas o discretas con o sin influencia temporal. Comúnmente se considera a $\mathbf{Y}_t$ como variable continua en un tiempo t que toma valores enteros $\mathbb{Z}$. Revisar Box, G. E., Jenkins, G. M., Reinsel, G. C., Ljung, G. M. (2015) para mas información.

Modelos lineales generalizados Los modelos lineales generalizados son una generalización del análisis de regresión lineal que surge de un arreglo de una combinación lineal de las covariables y dos funciones G, V y una constante C.

Donde:

$$G(\mu_i) = \sum_{j=0}^{p} \beta_j x_{ij}$$

y

$$Var[Y_i] = CV(\mu)$$

con

$$\mathbb{E}[Y_i] = \mu_i.$$

Donde la función G describe como interactúan la respuesta media de la variable dependiente con la combinación lineal de las covariables y V relaciona la varianza con la respuesta media. C es una constante de dispersión. Véase McCullagh, P., Nelder, J. A. (1989).

El investigador debe ponderar las ventajas y desventajas para decidir la técnica adecuada para modelar sus datos.

3. Supuestos

A continuación se presentan los supuestos del modelo de regresión lineal múltiple.

1. La variable respuesta es la imagen de una función lineal $R(\mathbf{x}) : \mathbb{R}^p \to \mathbb{R}$ de las variables regresoras y una variable aleatoria ε (p es el número de regresores).

$$Y_i = R(x_{i1}, \ldots, x_{ip}, \varepsilon_i) = \sum_{j=0}^{p} \beta_j x_{ij} + \varepsilon_i$$

con

$$x_{i0} = 1.$$

Usando notación matricial se expresa como:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}. \tag{1}$$

Donde:

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}_{n\times 1}$$

es el vector de observaciones,

$$\mathbf{X} = \begin{pmatrix} 1 & x_{1,2} & \dots & x_{1,p} \\ 1 & x_{2,2} & \dots & x_{2,p} \\ \vdots & \vdots & \dots & \vdots \\ 1 & x_{n,2} & \dots & x_{n,p} \end{pmatrix}_{n\times(p+1)}$$

es una matriz constante,

$$\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}_{(p+1)\times 1}$$

es el vector de parámetros y

$$\boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}_{n\times 1}$$

es un vector aleatorio.

2. Los regresores $\mathbf{X}_1, \dots, \mathbf{X}_p$ son variables independientes y cada observación $x_{i,j}$ es una constante conocida.
3. El vector de parámetros $\boldsymbol{\beta}$ es un vector constante desconocido.
4. El vector de residuales $\boldsymbol{\varepsilon}$ proviene de una distribución normal multivariada $N(0, \sigma^2\mathbf{I}_{nxn})$ con σ una constante positiva e $\mathbf{I}_{nxn}$ la matriz identidad de tamaño n.

De los supuestos, se puede demostrar que:

$$\mathbb{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$$

y

$$Var[\mathbf{Y}] = \sigma^2\mathbf{I}.$$

4. Estimación

Para llegar a un primer estimador de $\mathbf{Y}$ (se escribirá como $\widehat{\boldsymbol{Y}}$) se supondrá que se tienen identificadas todas las variables que intervienen en la variable dependiente (que además están incluidas en los datos) y que se cuenta con mas observaciones que variables.

Como se veía, uno de los supuestos es que la variable respuesta ($\mathbf{Y}$) es lineal en función de las covariables, luego al calcular su esperanza se observa que queda en términos de los estimadores de los parámetros ($\widehat{\boldsymbol{\beta}}$), por lo que encontrarlos sería un primer paso.

$$\mathbb{E}[\mathbf{Y}] = \widehat{\boldsymbol{Y}} = \mathbf{X}\widehat{\boldsymbol{\beta}} = \sum_{j=0}^{p} \widehat{\beta}_j x_j.$$

4.1. Estimación de los parámetros del modelo (β)

Intuitivamente, un buen modelo haría que la distancia entre los valores observados Y_i y los esperados ($\widehat{Y}_i$) no sea muy grande, pero hay muchas formas de definir ésta distancia, la manera más sencilla (ya que es diferenciable) para calcular una estimación de $\boldsymbol{\beta}$ es definiendo la misma como la suma de los cuadrados de las diferencias entre cada observación y el hiperplano definido por el estimador.

$$\sum \widehat{\varepsilon}_i^2 = \sum (Y_i - \widehat{Y}_i)^2. \tag{2}$$

Si se toma como criterio la suma de cuadrados de lo residuales o errores (Ecuación (2), se abreviará como SCE), entonces se encontrará $\widehat{\boldsymbol{\beta}}$ minimizando ésta expresión. La optimización se obtendrá derivando respecto a cada $\widehat{\beta}_i$.

$$SCE = \sum \varepsilon_i^2 = \sum (Y_i - \widehat{Y}_i)^2 = \sum (Y_i - \beta_0 - \beta_1 x_{i,1} \cdots - \beta_p x_{i,p})^2. \tag{3}$$

Derivando la Ecuación (2) respecto a cada β_j:

$$\frac{\partial SCE}{\partial \beta_j} = \sum_{i=1}^{n} 2(Y_i - \beta_0 x_{i,0} - \beta_1 x_{i,1} \cdots - \beta_p x_{i,p})(-x_{i,j}) \tag{4}$$

con $x_{i,0} = 1$.

Seguido de igualar a cero,

$$\frac{\partial SCE}{\partial \beta_j} = \sum_{i=1}^{n} (Y_i - \beta_0 x_{i,0} - \beta_1 x_{i,1} \cdots - \beta_p x_{i,p}) x_{i,j} = 0. \tag{5}$$

Si $\mathbf{X}_{i,*}$ representa el vector fila número i y $\mathbf{X}_{*,j}$ el vector columna número j la expresión anterior puede reescribirse como:

$$0 = \sum_{i=1}^{n} x_{i,j}(Y_i - \mathbf{X}_{i,*}\boldsymbol{\beta}) = \mathbf{X}_{*,j}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}).$$

Y como se repite para cada j, se puede visualizar como:

$$\mathbf{0}_{(p+1)\times 1} = \mathbf{X}'(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}).$$

Entonces

$$\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y}. \tag{6}$$

La Ecuación (6) es conocida como las ecuaciones normales de regresión en su forma matricial.

A continuación se muestran las constantes de estas ecuaciones:

$$\begin{pmatrix} n & \sum_{i=1}^n x_{1,i} & \cdots & \sum_{i=1}^n x_{p,i} \\ \sum_{i=1}^n x_{1,i} & \sum_{i=1}^n x_{1,i}^2 & \cdots & \sum_{i=1}^n x_{1,i}x_{1,p} \\ \vdots & \vdots & \cdots & \vdots \\ \sum_{i=1}^n x_{p,i} & \sum_{i=1}^n x_{p,i}x_{1,i} & \cdots & \sum_{i=1}^n x_{p,i}^2 \end{pmatrix} \boldsymbol{\beta} = \begin{pmatrix} \sum_{i=1}^n Y_i \\ \sum_{i=1}^n x_{1,i}Y_i \\ \vdots \\ \sum_{i=1}^n x_{p,i}Y_i \end{pmatrix}.$$

De la Ecuación (6) se desprende el estimador por mínimos cuadrados para $\widehat{\boldsymbol{\beta}}$ siempre que la matriz $(\mathbf{X}'\mathbf{X})^{-1}$ exista (que se garantiza si se cumple el supuesto de no colinealidad).

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}. \tag{7}$$

Para comprobar que $\widehat{\boldsymbol{\beta}}$ es el punto que minimiza (2) se usará el criterio de la segunda derivada. De (4) se obtiene la segunda derivada parcial respecto del parámetro j:

$$\frac{\partial^2 SCE}{\partial \beta_j^2} = \sum_{i=1}^{n} 2(-x_{i,j})(-x_{i,j}) > 0.$$

Observando que es mayor a cero ya que no se cuenta con variables constantes y por la desigualdad de Cauchy–Schwarz también se cumple que:

$$\frac{\partial^2 SCE}{\partial \beta_j^2}\frac{\partial^2 SCE}{\partial \beta_k^2} - \frac{\partial SCE}{\partial \beta_j \partial \beta_k}^2 = (\sum_{i=1}^{n} 2x_{i,j}^2)(\sum_{i=1}^{n} 2x_{i,k}^2) - (\sum_{i=1}^{n} 2x_{i,j}x_{i,k})^2 > 0.$$

Por lo que se confirma que se trata de un punto mínimo. Ahora que se cuenta con el estimador, lo mas natural es querer conocer la esperanza y varianza, así que se calcularán.

- Cálculo de la esperanza

$$\begin{aligned}
\mathbb{E}[\widehat{\boldsymbol{\beta}}] &= \mathbb{E}[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}] \\
&= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{E}[\mathbf{Y}] \\
&= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{E}[\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}] \\
&= (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})\boldsymbol{\beta} \\
&= \boldsymbol{\beta}.
\end{aligned}$$

$$\therefore \quad \mathbb{E}[\widehat{\boldsymbol{\beta}}] = \boldsymbol{\beta}. \tag{8}$$

- Cálculo de la varianza

$$\begin{aligned}
Var[\widehat{\boldsymbol{\beta}}] &= \mathbb{E}[(\widehat{\boldsymbol{\beta}} - \mathbb{E}[\widehat{\boldsymbol{\beta}}])(\widehat{\boldsymbol{\beta}} - \mathbb{E}[\widehat{\boldsymbol{\beta}}])'] \\
&= \mathbb{E}[(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta})(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta})'] \\
&= \mathbb{E}[((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{\varepsilon})((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{\varepsilon})'] \\
&= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{E}[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}']\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\
&= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\sigma^2\mathbf{I})\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\
&= \sigma^2(\mathbf{X}'\mathbf{X})^{-1}.
\end{aligned}$$

$$\therefore \quad Var[\widehat{\boldsymbol{\beta}}] = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}. \tag{9}$$

Bien, ya se conoce la varianza, pero ¿puede haber otro estimador de menor varianza? El siguiente teorema da la respuesta para los estimadores lineales e insesgados.

4.2. Teorema Gauss-Márkov

El teorema de Gauss-Márkov indica que el estimador de mínimos cuadrados es el mejor estimador (por ser el de menor varianza) en la clase de los estimadores lineales e insesgados, es decir, es un método óptimo de aproximación de los parámetros del modelo.

Teorema 4.1. *Dadas las hipótesis del modelo de regresión lineal, el estimador de $\boldsymbol{\beta}$ por mínimos cuadrados ordinarios ($\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y}$) es el estimador lineal de varianza mínima.*

Demostración. Sea $\widetilde{\boldsymbol{\beta}}$ un estimador lineal e insesgado de $\boldsymbol{\beta}$, como este estimador es lineal debería tener una forma como la siguiente:

$$\widetilde{\boldsymbol{\beta}} = \mathbf{B}\mathbf{Y}.$$

Donde $\widetilde{\boldsymbol{\beta}}$ es un vector de $(p+1)\times 1$, $\mathbf{Y}$ es de tamaño $n\times 1$ y $\mathbf{B}$ es una matriz de tamaño $(p+1)\times n$.

Sacando la esperanza condicional a la matriz de diseño de cada lado de la anterior ecuación:

$$\begin{aligned}\mathbb{E}[\widetilde{\boldsymbol{\beta}}|\mathbf{X}] &= \mathbf{B}\mathbb{E}[\mathbf{Y}|\mathbf{X}]\\ &= \mathbf{B}\mathbb{E}[\mathbf{X}\boldsymbol{\beta}+\boldsymbol{\varepsilon}|\mathbf{X}]\\ &= \mathbf{B}\mathbf{X}\boldsymbol{\beta}+\mathbf{B}\mathbb{E}[\boldsymbol{\varepsilon}|\mathbf{X}]\\ &= \mathbf{B}\mathbf{X}\boldsymbol{\beta}.\end{aligned}$$

De estas líneas se puede concluir que $\mathbf{B}\mathbf{X} = \mathbf{I}$, ya que $\widetilde{\boldsymbol{\beta}}$ es insesgado.

Sin pérdida de generalidad es válido reescribir $\mathbf{B}$ como $\mathbf{B} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}$ para alguna matriz $\mathbf{D}$ de tamaño $(p+1)\times n$ (ya que $\mathbf{D}$ es capaz de contener $-(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$, además, con ésta notación $\mathbf{B}$ puede reducirse a la proyección del estimador de mínimos cuadrados cuando $\mathbf{D} = \mathbf{0}_{(p+1)\times n}$).

Del hecho de que $\mathbf{B}\mathbf{X} = \mathbf{I}$ y de la forma de $\mathbf{B}$ es posible ver que $[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}]\mathbf{X} = \mathbf{I}$ entonces $\mathbf{I} + \mathbf{D}\mathbf{X} = \mathbf{I}$, por lo que $\mathbf{D}\mathbf{X} = \mathbf{0}_{(p+1)\times(p+1)}$. Aplicando lo que se ha visto a la forma original de $\widetilde{\boldsymbol{\beta}}$:

$$\widetilde{\boldsymbol{\beta}} = \mathbf{B}\mathbf{Y} = \mathbf{B}(\mathbf{X}\boldsymbol{\beta}+\boldsymbol{\varepsilon}) = \boldsymbol{\beta} + \mathbf{B}\boldsymbol{\varepsilon}.$$

Luego $\widetilde{\boldsymbol{\beta}} - \boldsymbol{\beta} = \mathbf{B}\boldsymbol{\varepsilon}$, con esta información se puede calcular la matriz de varianzas-covarianzas.

$$\begin{aligned}\mathbb{E}[(\widetilde{\boldsymbol{\beta}}-\boldsymbol{\beta})(\widetilde{\boldsymbol{\beta}}-\boldsymbol{\beta})'|X] &= \mathbb{E}[(\mathbf{B}\boldsymbol{\varepsilon})(\mathbf{B}\boldsymbol{\varepsilon})'|\mathbf{X}]\\ &= \mathbb{E}[\mathbf{B}\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}'\mathbf{B}']\\ &= \mathbf{B}\mathbb{E}[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}']\mathbf{B}'\\ &= \mathbf{B}(\sigma^2\mathbf{I})\mathbf{B}'\\ &= \sigma^2\mathbf{B}\mathbf{B}'.\end{aligned}$$

Es decir $Var[\widetilde{\boldsymbol{\beta}}] = \sigma^2\mathbf{B}\mathbf{B}'$. Desarrollando la última expresión:

$$\begin{aligned}\mathbf{B}\mathbf{B}' &= [(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}][(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}]'\\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} + (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{D}' + \mathbf{D}\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} + \mathbf{D}\mathbf{D}'.\end{aligned}$$

Y como se mostró $\mathbf{DX} = \mathbf{0}_{(p+1)\times(p+1)}$ (también $\mathbf{X'D'} = \mathbf{0}_{(p+1)\times(p+1)}$), por lo tanto $\mathbf{BB'} = (\mathbf{X'X})^{-1} + \mathbf{DD'}$. Por lo tanto:

$$Var[\widetilde{\boldsymbol{\beta}}] = Var[\widehat{\boldsymbol{\beta}}] + \sigma^2\mathbf{DD'}.$$

Como $\mathbf{DD'}$ es definida no negativa, entonces la expresión anterior es mínima en cada una de las estradas de la matriz $Var[\widetilde{\boldsymbol{\beta}}]$ cuando $\mathbf{D} = \mathbf{0}_{(p+1)\times 1}$. Y se expresa como $Var[\widehat{\boldsymbol{\beta}}] \preceq Var[\widetilde{\boldsymbol{\beta}}]$. □

4.3. Estimación de la varianza de los residuales (σ^2)

Antes de dar el estimador de la varianza, un poco de notación.

Se nombrará SCR a la suma de los cuadrados de las diferencias entre el valor del hiperplano y la media simple de la variable dependiente, también se referirá a esto como suma de cuadrados de la regresión.

$$SCR = \sum_{i=1}^{n}(\widehat{Y}_i - \bar{Y})^2. \tag{10}$$

Se reescribe la Ecuación (3) para poner énfasis en SCE

$$SCE = \sum_{i=1}^{n}\widehat{\varepsilon}_i^2 = \sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2. \tag{11}$$

Se llamará suma de cuadrados totales (SCT) a la suma de los cuadrados de las diferencias de las observaciones de la variable dependiente Y y su media.

$$SCT = \sum_{i=1}^{n}(Y_i - \bar{Y})^2.$$

Usando la notación anterior, se puede descomponer la suma de cuadrados totales en la suma de los cuadrados de la regresión más la de los residuales.

$$SCT = SCR + SCE. \tag{12}$$

La ecuación anterior expresa que la SCT (que es proporcional al estimador de la varianza de Y) se puede descomponer como la suma de los elementos, donde SCR representa la parte que se puede explicar a partir el modelo y la SCE como su complemento. Ahora se derivará (12):

$$\begin{aligned}
SCT &= \sum_{i=1}^{n}(Y_i - \overline{Y})^2 \\
&= \sum_{i=1}^{n}[(Y_i - \widehat{Y}_i) + (\widehat{Y}_i - \overline{Y})]^2 \\
&= \sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2 + \sum_{i=1}^{n}2(Y_i - \widehat{Y}_i)(\widehat{Y}_i - \overline{Y}) + \sum_{i=1}^{n}(\widehat{Y}_i - \overline{Y})^2 \\
&= SCE + 2\sum_{i=1}^{n}\widehat{\varepsilon}_i(\widehat{Y}_i - \overline{Y}) + SCR \\
&= SCE + 2(\sum_{i=1}^{n}\widehat{\varepsilon}_i\widehat{Y}_i - \overline{Y}\sum_{i=1}^{n}\widehat{\varepsilon}_i) + SCR.
\end{aligned}$$

Sólo falta mostrar que $(\sum_{i=1}^n \widehat{\varepsilon}_i \widehat{Y}_i - \overline{Y}\sum_{i=1}^n \widehat{\varepsilon}_i) = 0$, y esto se cumple ya que $\sum_{i=1}^n \widehat{\varepsilon}_i \widehat{Y}_i = 0$ y $\sum_{i=1}^n \widehat{\varepsilon}_i = 0$.

$\left(\sum_{i=1}^n \widehat{\varepsilon}_i = 0\right)$ La demostración de este hecho es inmediato pues es la primera ecuación normal ($\frac{\partial SCE}{\partial \beta_0} = 0$).

$\left(\sum_{i=1}^n \widehat{\varepsilon}_i \widehat{Y}_i = 0\right)$ Se probará la igualdad usando notación matricial.

$$\begin{aligned}
\sum_{i=1}^n \widehat{\varepsilon}_i \widehat{Y}_i &= \widehat{\boldsymbol{\varepsilon}}' \widehat{\mathbf{Y}} \\
&= \widehat{\boldsymbol{\varepsilon}}' (\mathbf{X}\widehat{\boldsymbol{\beta}}) \\
&= (\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})' \mathbf{X}\widehat{\boldsymbol{\beta}} \\
&= (\mathbf{Y}' - \widehat{\boldsymbol{\beta}}'\mathbf{X}')\mathbf{X}\widehat{\boldsymbol{\beta}} \\
&= (\mathbf{Y}'\mathbf{X} - \widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{X})\widehat{\boldsymbol{\beta}} \\
&= (\mathbf{X}'\mathbf{Y} - \mathbf{X}'\mathbf{X}\widehat{\boldsymbol{\beta}}')\widehat{\boldsymbol{\beta}} \\
&= 0 \qquad\qquad (\text{pues } \mathbf{X}'\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}'\mathbf{Y}).
\end{aligned}$$

Las expresiones de las sumas de cuadrados en notación matricial son las siguientes:

$$SCR = \widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{Y}' - \frac{1}{n}\mathbf{Y}'\mathbf{1}_{n\times 1}\mathbf{1}'_{n\times 1}\mathbf{Y}. \tag{13}$$

$$SCE = \widehat{\boldsymbol{\varepsilon}}'\widehat{\boldsymbol{\varepsilon}} = (\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}). \tag{14}$$

$$SCT = \mathbf{Y}'\mathbf{Y} - \frac{1}{n}\mathbf{Y}'\mathbf{1}_{n\times 1}\mathbf{1}'_{n\times 1}\mathbf{Y}. \tag{15}$$

Donde $\mathbf{1}_{n\times 1} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{n\times 1.}$

Para estimar los residuales ε se usa el estimador mas natural, que es la diferencia entre el valor observado y el plano estimado $\widehat{\varepsilon}_i = Y_i - X_i\widehat{\beta}$.

Si se pensara usar el estimador de la varianza por el método de momentos $\frac{\sum_{i=1}^n \varepsilon_i^2}{n}$ se obtiene un estimador sesgado (que coincide con el de máxima verosimilitud), pero a partir de este se avanza a uno insesgado.

$$MSE = \frac{SCE}{n-p-1}.$$

$$MSE = \frac{\sum_{i=1}^n \varepsilon_i^2}{n-p-1} = \frac{(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})}{n-p-1} = \frac{SCE}{n-p-1}.$$

Para demostrar que $\frac{\sum_{i=1}^n \varepsilon_i^2}{n}$ es un estimador con sesgo, se calculara el valor esperado de $(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})$.

Se llamará $\mathbf{P} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$. Es fácil comprobar que $\mathbf{P}$ una matriz idempotente y $\mathbf{P} = \mathbf{P}' = \mathbf{P}^2$.

$$\begin{aligned}(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}) &= (\mathbf{Y} - \mathbf{PY})'(\mathbf{Y} - \mathbf{PY}) \\ &= \mathbf{Y}'\mathbf{Y} - \mathbf{Y}'\mathbf{PY} - (\mathbf{PY})'\mathbf{Y} + \mathbf{PYPY} \\ &= \mathbf{Y}'\mathbf{Y} - \mathbf{Y}'\mathbf{PY} - \mathbf{Y}'\mathbf{P}'\mathbf{Y} + \mathbf{Y}'\mathbf{P}'\mathbf{PY} \\ &= \mathbf{Y}'\mathbf{Y} - \mathbf{Y}'\mathbf{XPY} \\ &= \mathbf{Y}'(\mathbf{I} - \mathbf{P})\mathbf{Y}.\end{aligned}$$

Dado que

$$\begin{aligned}(\mathbf{I} - \mathbf{P})\mathbf{Y} &= (\mathbf{I} - \mathbf{P})(\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}) \\ &= \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} + \mathbf{PX}\boldsymbol{\beta} + \mathbf{P}\boldsymbol{\varepsilon} \\ &= \mathbf{IX}\boldsymbol{\beta} + \mathbf{I}\boldsymbol{\varepsilon} - \mathbf{PX}\boldsymbol{\beta} - \mathbf{P}\boldsymbol{\varepsilon} \\ &= \mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\beta} + \mathbf{I}\boldsymbol{\varepsilon} - \mathbf{P}\boldsymbol{\varepsilon} \\ &= (\mathbf{I} - \mathbf{P})\boldsymbol{\varepsilon}.\end{aligned} \tag{16}$$

Y también es válida la ecuación transpuesta:

$$\mathbf{Y}'(\mathbf{I} - \mathbf{P}) = \boldsymbol{\varepsilon}'(\mathbf{I} - \mathbf{P}).$$

De lo anterior se concluye que:

$$(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}) = \boldsymbol{\varepsilon}'(\mathbf{I} - \mathbf{P})\boldsymbol{\varepsilon}.$$

De esto se desprende lo siguiente:

$$\mathbb{E}[(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})] = \mathbb{E}[\boldsymbol{\varepsilon}'\boldsymbol{\varepsilon}] - \mathbb{E}[\boldsymbol{\varepsilon}'\mathbf{P}\boldsymbol{\varepsilon}]. \tag{17}$$

Evaluando los términos del lado derecho de la ecuación anterior

$$\mathbb{E}[\boldsymbol{\varepsilon}'\boldsymbol{\varepsilon}] = \sum_{i=1}^{n} \mathbb{E}[\varepsilon_i^2] = n\sigma^2. \tag{18}$$

$$\begin{aligned}Traza\{\mathbb{E}[\boldsymbol{\varepsilon}'\mathbf{P}\boldsymbol{\varepsilon}]\} &= \mathbb{E}[Traza\{\boldsymbol{\varepsilon}'\mathbf{P}\boldsymbol{\varepsilon}\}] \\ &= \mathbb{E}[Traza\{\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}'\mathbf{P}\}] \\ &= Traza\{\mathbb{E}[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}']\mathbf{P}\} \\ &= Traza\{\sigma^2\mathbf{P}\} \\ &= \sigma^2 Traza\{\mathbf{P}\} \\ &= \sigma^2 p.\end{aligned} \tag{19}$$

Lo último se debe a que $Traza(\mathbf{P}) = Traza(\mathbf{I}) = p$. Aplicando desde la Ecuación (18) a la Ecuación (17) se obtiene:

$$\mathbb{E}\{(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})\} = \sigma^2(n - p - 1). \tag{20}$$

De (20) se deduce el estimador insesgado.

4.4. Estimación por máxima verosimilitud

Hasta el momento el texto se ha guiado por argumentos geométricos para deducir los estimadores del modelo de regresión lineal y se mostró que es un buen estimador por el Teorema Gauss-Márkov. Ahora el interés es encontrar los estimadores de máxima verosimilitud.

Se partirá de la densidad conjunta de los residuales (bajo los supuestos de independencia de los residuales y normalidad).

$$f(\boldsymbol{\varepsilon},\boldsymbol{\beta},\sigma^2) = (\frac{1}{2\pi\sigma^2})^{n/2}\exp(-\frac{1}{2\sigma^2}\boldsymbol{\varepsilon}'\boldsymbol{\varepsilon}).$$

O equivalentemente:

$$f(\boldsymbol{\varepsilon},\boldsymbol{\beta},\sigma^2) = (\frac{1}{2\pi\sigma^2})^{n/2}\exp(-\frac{1}{2\sigma^2}(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})). \tag{21}$$

Por definición, los estimadores de máxima verosimilitud para $\boldsymbol{\beta}$ y σ^2 son los puntos ($\widehat{\boldsymbol{\beta}}$ y $\widehat{\sigma}^2$ respectivamente) tales que maximizan (21), que por lo común se encuentran usando las derivadas parciales del logaritmo de tal función, ya que el logaritmo respeta los puntos máximos (al ser creciente) simplificando los cálculos y ambas funciones son continuas y diferenciables. Procediendo:

$$\frac{\partial log(f)}{\partial\boldsymbol{\beta}} = \frac{\partial(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = 0.$$

Observando la expresión anterior se puede ver que se han encontrado las ecuaciones normales.

$$\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y}.$$

Por lo tanto el estimador de $\boldsymbol{\beta}$ es el mismo que por mínimos cuadrados. Luego, para estimar σ^2 se toma la derivada parcial respecto de este parámetro.

$$\frac{\partial log(f)}{\partial\sigma^2} = -\frac{n}{2\sigma^2} + \frac{n}{2\sigma^4}(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}) = 0.$$

Así, el estimador de σ^2 se expresa como:

$$\widehat{\sigma}^2 = \frac{(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})}{n}.$$

5. Validación del modelo de regresión lineal

Después de conocer un buen estimador y su varianza, es deseable tener evidencia de que se cumplen los supuestos del modelo. Es decir, se pretende proporcionar validaciones gráficas de normalidad de los residuales, pruebas de su independencia y homocedasticidad, además de poder verificar que las variables que se incluyen en el modelo realmente influyen en el proceso que genera los datos. En este sentido, se han desarrollado técnicas para brindar tales evidencias. Se mostrarán algunas.

5.1. Independencia de los residuales

Una de las consecuencias de la estimación de los parámetros vía mínimos cuadrados ordinarios es la dependencia lineal de los residuales.

$$\begin{aligned}\sum_{i=1}^{n}\hat{\varepsilon}_i &= \sum_{i=1}^{n}(Y_i-\beta_0-x_{1,i}\beta_1-\cdots-x_{p,i}\beta_p)\\ &= \sum_{i=1}^{n}Y_i-n\beta_0-\beta_1\sum_{i=1}^{n}x_{1,i}-\cdots-\beta_p\sum_{i=1}^{n}x_{p,i}\\ &= 0.\end{aligned} \tag{22}$$

La última ecuación se sigue de la condición expresada en (5), lo que prueba que cada uno de los residuales se puede obtener con el resto. Aun así, se trata de mantener controlada la dependencia verificando que no existe correlación serial de los residuales.

Prueba Durbin-Watson

La prueba Durbin-Watson se desarrolla para probar la dependencia entre $\hat{\varepsilon}_i$ y $\hat{\varepsilon}_{i-1}$.

$$H_0 : \varrho = 0 \text{ vs } H_a : \varrho \neq 0.$$

En esta prueba la hipótesis alternativa asegura que $\varepsilon_t = \varrho\varepsilon_{t-1} + \upsilon_t$. Donde

1. υ_t para $t = 1,\ldots,n$ son variables aleatorias independientes e idénticamente distribuidas que se generan a partir de una normal $N(0,\sigma^2)$, para una constante σ^2 positiva.

2. $|\varrho| < 1$, ϱ es llamada autocorrelación a un paso de los residuales.

3. El orden cronológico t es el orden de los residuales, es decir, de acuerdo al orden con el que se encuentran las observaciones originales.

Estadístico de prueba

$$d = \frac{\sum_{i=1}^{n}(\varepsilon_i-\varepsilon_{i-1})^2}{\sum_{i=1}^{n}\varepsilon_i^2}.$$

Para esta prueba se han calculado tablas de valores críticos d_L y d_U que auxilian a delimitar la región de no rechazo (si $d_U < d < 4-d_U$ entonces no se rechaza la hipótesis nula). Si se rechazara la hipótesis nula, implicaría que el modelo de regresión no ha terminado de explicar el componente sistemático que existe entre la variable respuesta y sus regresores. Otra posible consecuencia es la no linealidad, en tal caso, se debe proponer alguna transformación para resolver el problema (Durbin, J., Watson, G.S. [1950]).

5.2. Normalidad de los residuales

La mayor parte de las pruebas de hipótesis de normalidad como Kolmogórov-Smirnov, Shapiro-Wilk, Jarque–Bera, o de Geary suponen que la muestra es independiente e idénticamente distribuida, y como se muestra en (22) esto no ocurre, así, lo mas apropiado es comprobar la normalidad a través de la curtosis, un qqplot o un histograma.

Histograma

Un histograma es una gráfica de barras que representa la distribución de una muestra a lo largo de su rango, agrupando los datos por clases. Cada clase estará representada por un rectángulo, donde comparten el mismo tamaño de base y la altura es proporcional a la frecuencia de la clase. Se dirá que un elemento está en determinada clase si su valor se encuentra en la base del rectángulo. Es decir:

Sea

- $X_1,\ldots,X_n$ es una muestra.
- $R = Max\{X_1,\ldots,X_n\} - Min\{X_1,\ldots,X_n\}$ el rango muestral.
- K el número de clases, se sugieren las siguientes opciones:
  - $K = \sqrt{n}$.
  - Fórmula de Sturges : $K = 1 + \log_2(n)$.
  - Regla de Scott: $K = (2n)^{1/3}$.
  - Regla de Freedman–Diaconis: $K = 2\frac{RIQ}{\sqrt[3]{n}}$ donde RIQ es el rango intercuartil muestral.
-

$$C_j = [Min\{X_1,\ldots,X_n\} + (j-1)\frac{R}{K}, Min\{X_1,\ldots,X_n\} + j\frac{R}{K}] \quad \text{para } j = 1,\ldots,K.$$

Entonces el histograma de frecuencias relativas es la representación de la función:

$$f(z) = \frac{\sum_{i=1}^{n}(1(x_i)_{C_j})}{n} \qquad \text{si } z \in C_j.$$

Y el histograma de frecuencias absolutas:

$$f(z) = \sum_{i=1}^{n}(1(x_i)_{C_j}) \qquad \text{si } z \in C_j.$$

Donde $1(x_i)_{C_j}$ es la indicadora de x_i sobre el conjunto C_j.

Se comprobará si una muestra proviene de una distribución normal, si el observador considera que no existe una gran diferencia entre el histograma de frecuencias relativas y la gráfica (especialmente en la moda y las colas) de la densidad de una normal $N(\hat{\mu},\hat{\sigma}^2)$ (para $\widehat{\mu}$ y $\widehat{\sigma}^2$ convenientes, en el caso de los residuales se puede considerar $\mu = 0$ y $\hat{\sigma}^2$ como la máxima verosímil o bien, considerarla $\hat{\sigma}^2 = 1$ si se estandarizan los residuales).

En las siguientes imágenes se muestran dos histogramas comparados contra la densidad de una normal:

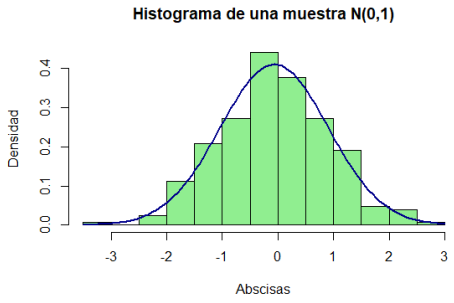

Figura 1: Histograma de frecuencias relativas de una muestra normal estándar de tamaño 250 calculado por la regla de Sturges contra la densidad de una normal $N(\hat{\mu}, \hat{\sigma}^2)$ para los parámetros máximo verosímiles.

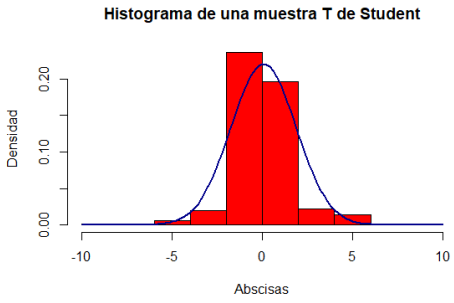

Figura 2: Histograma de frecuencias relativas de una muestra T de Student con 3 grados de libertad de tamaño 250 calculado por la regla de Sturges contra la densidad de una normal $N(\hat{\mu}, \hat{\sigma}^2)$ para los parámetros máximo verosímiles.

QQ-plot

Si F_X y G_Y son funciones de distribución cualesquiera y sea u tal que $0 < u < 1$, se nombrará como F^* a la función tal $F(u)^* := Inf\{x : F(x) \geq u\}$, y a la gráfica formada por los las tuplas $(F_X(u)^*, G_Y(u)^*)$ para toda $u \in (0,1)$ como QQ-plot de G sobre F.

Cuando Y es tal que $Y \sim aX + b$, es decir X está correlacionada con Y, entonces el QQ-plot muestra una relación lineal, ya que:

$$\begin{aligned} G_Y^*(u) &= Inf\{x : G_Y(x) \geq u\} \\ &= Inf\{x : F_X(\frac{x-b}{a}) \geq u\} \\ &= Inf\{ax + b : F_X(x) \geq u\} \\ &= a\ Inf\{x : F_X(x) \geq u\} + b \\ &= a\ F_X^*(u) + b. \end{aligned} \tag{23}$$

Esto ya que $G_Y(x) = P(Y \leq x) = P(aX + b \leq x) = P(X \leq \frac{x-b}{a}) = F_X(\frac{x-b}{a})$.

Si quisiera comparar la distribución de los dos muestras $X_1, \ldots, X_n$ y $Y_1, \ldots, Y_n$ (en principio del mismo tamaño), se pueden formar tuplas con los estadísticos de orden $(X_{(i)}, Y_{(i)})$ para $i = 1, \ldots, n$, con la idea de que si la distribución de las variables dependen linealmente una de la otra, entonces los estadísticos ($X_{(i)}$ y $Y_{(i)}$) formarían una recta, ya que la fracción de la muestra de los i-ésimos estadísticos debería ser la misma en cada muestra.

Normal estándar contra normal estándar En la siguiente imagen se ve un QQ-plot de dos muestras que provienen de la misma distribución.

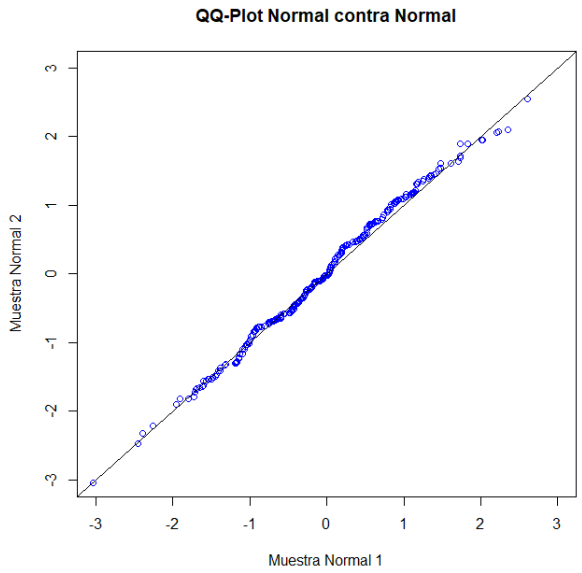

Figura 3: QQ-plot de dos muestras normales estándar de tamaño 250.

Muestra normal estándar contra sus cuantiles esperados El siguiente QQ-plot exhibe la relación de una muestra normal estándar contra los cuantiles esperados.

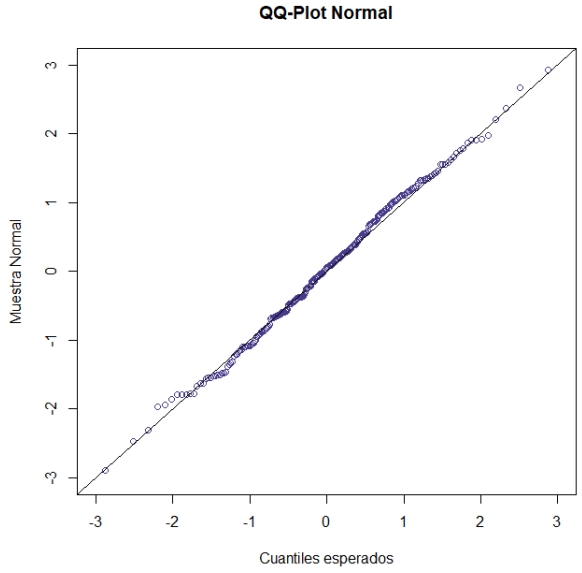

Figura 4: QQ-plot de una muestras normales estándar de tamaño 250 contra sus cuantiles esperados.

T de Student contra normal estándar Aquí se ven dos muestras de diferentes distribuciones que no forman una relación lineal. Se puede comprobar la normalidad de

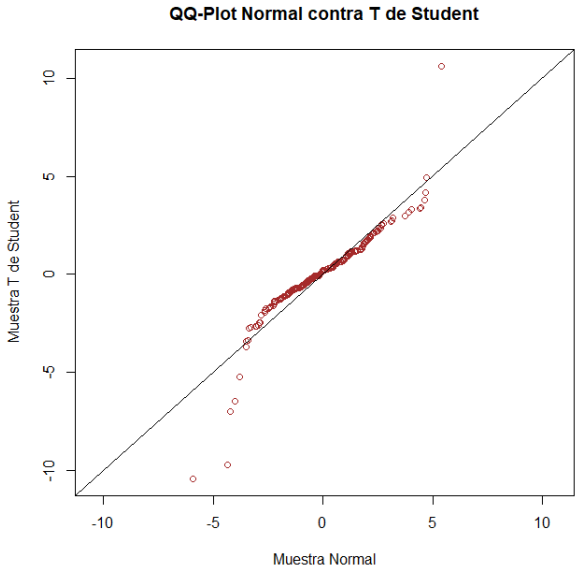

Figura 5: QQ-plot de una muestra t con 2 grados de libertad contra una muestra de una normal estándar, ambas de tamaño 250.

$X_1, \ldots, X_n$ formando las tuplas $(X_{(i)}, \mathbb{E}[X_{(i)}])$ donde $\mathbb{E}[X_{(i)}]$ es el valor esperado de la i-ésima estadística de orden de una normal $N(0,1)$, puesto que una combinación de normales es normal.

Box-Plot

Un Box-plot o diagrama de caja y brazos es una gráfica donde se visualiza los cuartiles, el máximo y el mínimo de una muestra. Es útil para mostrar asimetrías en las colas de la distribución así como la curtosis.

Pasos para crear un Box-plot Dada una muestra $X_1, \ldots, X_n$.

1. Ordenar la muestra $(X_{(1)}, \ldots, X_{(n)})$.
2. Calcular los primeros 3 cuartiles, el máximo y mínimo de la distribución. Donde los cuartiles se calculan de la siguiente forma:

   $$F^{-1}(p) = inf\{x \in R : p \leq F(x)\}.$$

   Con F la función de distribución muestral y $p = 0.25, 0.5, 0.75$.
3. Sobre una recta, seleccionar un segmento como unidad de medida y marcar sobre ésta las estadísticas calculadas en el paso anterior.
4. Trazar un segmento de recta perpendicular (de tamaño igual a la unidad u otro que se considere conveniente) sobre los cuartiles y dos rectas paralelas al eje que formen un rectángulo que tenga por lados primer y tercer cuartil.

Simetría de la distribución Cuando una distribución es simétrica (o asimétrica), su box-plot hereda esta propiedad mostrando una proporción insesgada (sesgada a la izquierda o derecha) de los cuartiles. A continuación se muestra el histograma y el Box-plot de una muestra Ji-cuadrada (sesgada a la izquierda):

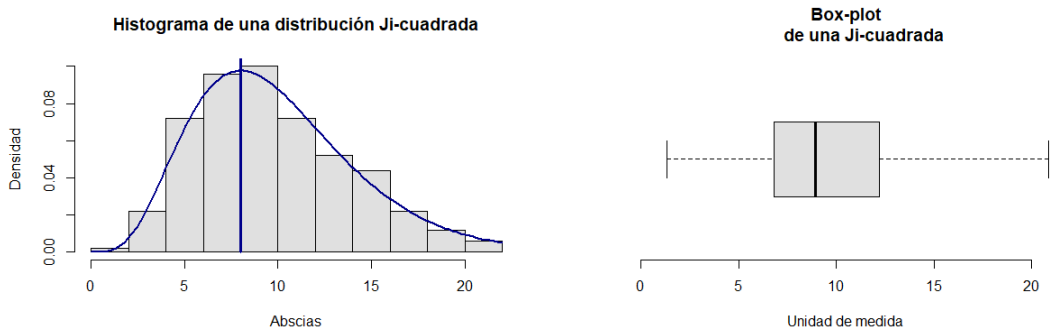

Figura 6: Histograma contra Box-plot de una muestra Ji-cuadrada de 10 grados de libertad de tamaño 250.

Curtosis La curtosis es una medida que sirve para comparar la cantidad de varianza que es producida por cuan alejadas están las observaciones de su media. Usando integración por partes, se puede mostrar que $\mathbb{E}[Z^4] = 3$ para $Z \sim N(0,1)$. Ahora, para cualquier variable aleatoria X con $\mathbb{E}[X^4] < \text{ínf}$, $\mathbb{E}[X] = \mu$ y $\sigma > 0$, se define la curtosis de X como:

$$\gamma_2(X) := \frac{\mathbb{E}[(X-\mu)^4]}{\sigma^2} - 3.$$

Entonces, a medida que la curtosis aumenta los datos son mas próximos a su media. Si la curtosis disminuye, los datos se agrupan hacia las colas de la distribución. A partir de que la única distribución con curtosis 0 es una normal estándar, se clasifican las distribuciones de la siguiente forma:

- Leptocúrtica, cuando $\gamma_2(X) > 0$.
- Mesocúrtica, cuando $\gamma_2(X) = 0$.
- Platicúrtica, cuando $\gamma_2(X) < 0$.

Dicho lo anterior, de una muestra normal se espera un box-plot que muestre una distribución mesocúrtica, simétrica respecto de la mediana y que la distancia entre el primer y segundo cuartil sea aproximadamente la misma que la del segundo al tercero. Aplica lo mismo para las observaciones máxima y mínima, se espera sean simétricas respecto de la mediana y la diferencia con el primer y tercer cuartil (respectivamente) sea la misma (aproximadamente).

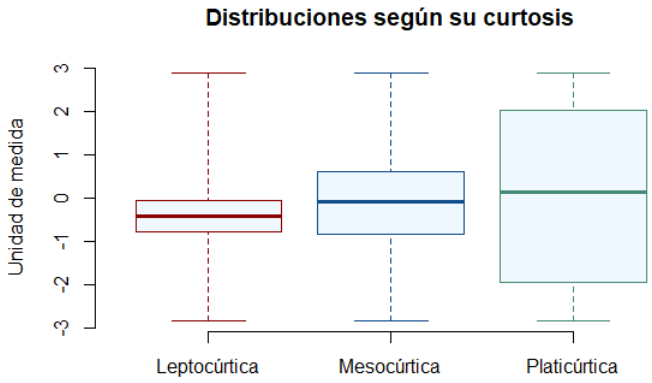

Figura 7: Box-plot como herramienta para comparar la curtosis.

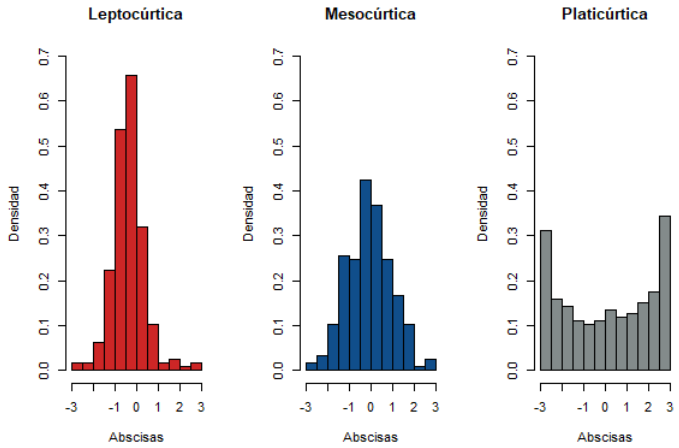

Figura 8: Histogramas según su curtosis

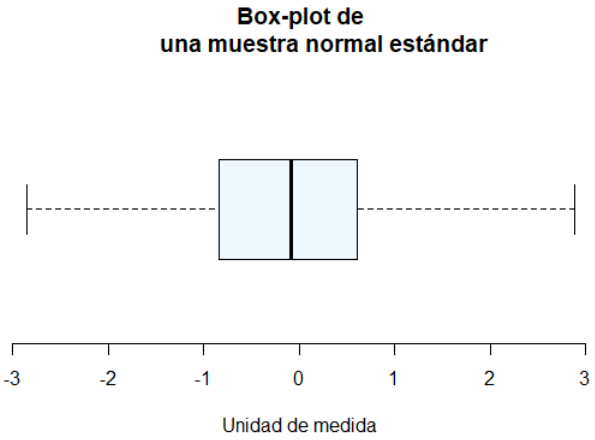

Figura 9: Box-plot de una muestra normal estándar de tamaño 250.

5.3. Homocedasticidad

Para probar el supuesto de homocedasticidad es posible auxiliarse de la prueba de Levene. Esta prueba se usa para demostrar la igualdad de las varianzas de $m \geq 2$ poblaciones normales $X_i \sim N(\mu_i, \sigma_i^2)$.

Prueba de Levene

$$H_0 : \sigma_1^2 = \sigma_2^2 = \cdots = \sigma_m^2$$
$$\text{vs}$$
$$H_a : \exists k \in \{1, \ldots, m\} \text{ tal que } \sigma_k^2 \neq \sigma_j^2 \text{ para alguna } j \neq k.$$

Entonces, para aplicar la prueba se necesita formar m grupos (estos quedan al criterio del investigador). Cada grupo contiene n_i observaciones. El estadístico de prueba es el siguiente:

$$L = \frac{n-m}{m-1} \frac{\sum_{j=1}^{m} n_j (\bar{D}_j - \bar{D})^2}{\sum_{j=1}^{m} \sum_{i=1}^{n_j} (D_i j - \bar{D}_j)^2}.$$

Donde:

$$n = \sum_{j=1}^{m} n_j.$$
$$\bar{X}_j = \frac{\sum_{i=1}^{n_j} x_{ji}}{n_j}.$$
$$D_{ji} = |X_{ji} - \bar{X}_j|.$$
$$\bar{D}_j = \frac{\sum_{i=1}^{n_j} D_{ji}}{n_j}.$$

El estadístico L, es el estadístico de la tabla ANOVA aplicado a la variable D (distancia) $L \sim F(m-1, n-m)$. Se rechazará la hipótesis nula si para alguna partición de los residuales se consigue que $L > F_{m-1,n-m,1-\alpha}$.

Para solucionar el problema de la heterocedasticidad se puede intentar con una transformación de variables ó dar una estimación por mínimos cuadrados generalizados.

Existen otras pruebas que evitan el hecho de tener que formar grupos, por ejemplo, la prueba Breusch-Pagan o la de White, estas trabajan con regresiones auxiliares donde la variable respuesta son residuales y contra las variables explicativas del modelo original, si se encuentra una relación significativa entonces, se ha probado la heterocedasticidad de los residuales.

El contraste de Breusch-Pagan es sensible a la no normalidad de los residuales y se requiere que el investigador proponga el tipo de regresión auxiliar.Algunas opciones encontradas en la literatura son:
Breusch and Pagan, 1979:

$$e_i^2 = \alpha_0 + \alpha_1 X_{i1} + \cdots + \alpha_p X_{ip}.$$

Glesjer, 1969:

$$e_i = \alpha_0 + \alpha_1 X_{i1} + \cdots + \alpha_p X_{ip}.$$

Harvey 1976, Godfrey 1978:

$$\log(e_i^2) = \alpha_0 + \alpha_1 X_{i1} + \cdots + \alpha_p X_{ip}.$$

La prueba de White propone colocar en la regresión el producto de las variables explicativas a pares:

$$e_i^2 = \alpha_0 + \sum_{j=1}^{p} \alpha_j X_{ij} + \sum_{j=1}^{p} \alpha_{p+j} X_{ij}^2 + \sum_{j=1}^{p} \sum_{k=1}^{p} \alpha_h X_{ij} X_{ik}$$

donde $h(1,1) = 1$, $h(i,j) = h(i,j-1)+1$ y $h(i+1,j) = h(i,j+p+1)$ es el índice de las constantes. En ambos casos la hipótesis a contrastar es

$$H_0 : \alpha_1 = \cdots = \alpha_l = 0 \text{ vs } H_a : \exists k \in \{1,\ldots,l\} \text{ tal que } \alpha_k \neq 0.$$

En Breusch, T.S., Pagan, A.R. (1979) y White, H.(1980) se profundiza sobre el tema.

5.4. Hipótesis sobre los coeficientes

El modelo de regresión particionado

Antes de continuar se mostrarán algunos resultados que ayudaran a probar la relevancia de las variables en el modelo de regresión.

Supóngase que se cuenta con una partición $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \boldsymbol{\beta}_2]$ de los parámetros del modelo de regresión lineal múltiple, donde (sin pérdida de generalidad) $\boldsymbol{\beta}_1$ contiene las primeras p_1 variables y $\boldsymbol{\beta}_2$ las restantes, es decir, el modelo quedaría planteado como:

$$\mathbf{Y} = \mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_2\boldsymbol{\beta}_2 + \boldsymbol{\varepsilon}.$$

Con $\mathbf{X}_1$ y $\mathbf{X}_2$ matrices que contienen n observaciones de las primeras y últimas p_1 y p_2 variables respectivamente, $(\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2])$. Retomando las ecuaciones normales (6), estas se reescriben como:

$$\mathbf{X}_1'\mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_1'\mathbf{X}_2\boldsymbol{\beta}_2 = \mathbf{X}_1'\mathbf{Y}. \tag{24}$$

$$\mathbf{X}_2'\mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_2'\mathbf{X}_2\boldsymbol{\beta}_2 = \mathbf{X}_2'\mathbf{Y}. \tag{25}$$

Esto se obtiene de la siguiente manera:

$$\begin{aligned}
\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y} \iff & \quad (\mathbf{X}_1\mathbf{X}_2)'(\mathbf{X}_1\mathbf{X}_2) \quad \begin{pmatrix}\boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2\end{pmatrix} = (\mathbf{X}_1\mathbf{X}_2)'\mathbf{Y} \\
\iff & \quad \begin{pmatrix}\mathbf{X}_1' \\ \mathbf{X}_2'\end{pmatrix}(\mathbf{X}_1\mathbf{X}_2) \quad \begin{pmatrix}\boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2\end{pmatrix} = \begin{pmatrix}\mathbf{X}_1' \\ \mathbf{X}_2'\end{pmatrix}\mathbf{Y} \\
\iff & \quad \begin{pmatrix}\mathbf{X}_1'\mathbf{X}_1 & \mathbf{X}_1'\mathbf{X}_2 \\ \mathbf{X}_2'\mathbf{X}_1 & \mathbf{X}_2'\mathbf{X}_2\end{pmatrix} \begin{pmatrix}\boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2\end{pmatrix} = \begin{pmatrix}\mathbf{X}_1' \\ \mathbf{X}_2'\end{pmatrix}\mathbf{Y}.
\end{aligned}$$

Estimadores del modelo particionado

Como se vio anteriormente, el estimador de $\boldsymbol{\beta}$ por mínimos cuadrados es aquel que resuelve las ecuaciones normales, lo que se hará a continuación es mostrar la forma que adquiere con una partición de sus parámetros resolviendo el sistema de ecuaciones matriciales.

Multiplicando a (24) por $\mathbf{X}_2'\mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-1}$ igualando los términos constantes de $\boldsymbol{\beta}_1$ para ambas ecuaciones.

$$\mathbf{X}_2'\mathbf{X}_1\boldsymbol{\beta}_1 + [\mathbf{X}_2'\mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-1}]\mathbf{X}_1'\mathbf{X}_2\boldsymbol{\beta}_2 = \mathbf{X}_1'\mathbf{Y}. \tag{26}$$

Restando de (26) la ecuación (25) se elimina el término de β_1, quedando:

$$\{\mathbf{X}_2'\mathbf{X}_2 - \mathbf{X}_2'\mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2\}\boldsymbol{\beta}_2 = \mathbf{X}_2'\mathbf{Y} - \mathbf{X}_2'\mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{Y}. \tag{27}$$

Definiendo la proyección $\mathbf{P}_i$ de forma análoga a $\mathbf{P}$, $\mathbf{P}_i = \mathbf{X}_i(\mathbf{X}_i'\mathbf{X}_i)^{-1}\mathbf{X}_i'$ para $i = 1, 2$. Se puede reescribir la ecuación (27) como:

$$\{\mathbf{X}_2'\mathbf{X}_2 - \mathbf{X}_2'\mathbf{P}_1\mathbf{X}_2\}\boldsymbol{\beta}_2 = \mathbf{X}_2'\mathbf{Y} - \mathbf{X}_2'\mathbf{P}_1\mathbf{Y}.$$
$$\mathbf{X}_2'(\mathbf{I} - \mathbf{P}_1)\mathbf{X}_2\boldsymbol{\beta}_2 = \mathbf{X}_2'(\mathbf{I} - \mathbf{P}_1)\mathbf{Y}.$$

Entonces, el estimador por mínimos cuadrados para $\boldsymbol{\beta}_2$ queda como:

$$\hat{\boldsymbol{\beta}}_2 = \{\mathbf{X}_2'(\mathbf{I} - \mathbf{P}_1)\mathbf{X}_2\}^{-1}\mathbf{X}_2'(\mathbf{I} - \mathbf{P}_1)\mathbf{Y}. \tag{28}$$

Si de la ecuación (25) se despeja β_1 se obtiene su estimador en términos de $\hat{\boldsymbol{\beta}}_2$

$$\hat{\boldsymbol{\beta}}_1 = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'(\mathbf{Y} - \mathbf{X}_2\hat{\boldsymbol{\beta}}_2). \tag{29}$$

Algunos resultados de utilidad

La hipótesis que el modelo $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ tiene residuales $\boldsymbol{\varepsilon}$ normales, independientes e idénticamente distribuidos es de gran utilidad para desarrollar pruebas de hipótesis. A continuación algunos resultados de probabilidad que servirán posteriormente:

Sea $\mathbf{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ con media $\boldsymbol{\mu}$ y matriz de dispersión $\mathbf{D}(\mathbf{X}) = \boldsymbol{\Sigma}$.

Proposición 5.1. *Si $\boldsymbol{T}$ es una transformación tal que $\boldsymbol{T}\boldsymbol{\Sigma}\boldsymbol{T}' = \boldsymbol{I}$ y $\boldsymbol{T}\boldsymbol{T}' = \boldsymbol{\Sigma}^{-1}$ entonces $\boldsymbol{T}(\boldsymbol{X} - \boldsymbol{\mu}) \sim N(\boldsymbol{0}, \boldsymbol{I})$.*

Proposición 5.2. *Si $\boldsymbol{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ de dimensión n, entonces $(\boldsymbol{X} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{X} - \boldsymbol{\mu}) \sim \chi^2(n)$.*

Proposición 5.3. *Si $U \sim \chi^2(m)$ y $V \sim \chi^2(n)$ son dos variables independientes que distribuyen Ji-cuadrada de m y n grados de libertad respectivamente, entonces, la suma distribuye Ji-cuadrada de $m + n$ grados de libertad $(U + V) \sim \chi^2(m + n)$.*

Proposición 5.4. *Si $U \sim \chi^2(m)$ y $V \sim \chi^2(n)$ son dos variables independientes que distribuyen Ji-cuadrada de m y n grados de libertad respectivamente, entonces, la división de las variables divididas por sus grados de libertad distribuye como una F de m y n grados de libertad $F = \frac{U/m}{V/n} \sim F(m, n)$.*

Proposición 5.5. *Si $z \sim N(0, 1)$ y $v \sim \chi^2(n)$ son dos variables independientes, entonces la transformación $t = \frac{z}{\sqrt{v/n}}$ tiene una distribución T de Student de n grados de libertad $t \sim t(n)$.*

Distribuciones relacionadas con los coeficientes del plano de regresión

En esta parte del texto se tratará algunas distribuciones que están relacionadas al estimador $\widehat{\boldsymbol{\beta}}$.

Sabiendo que una combinación lineal de variables aleatorias normales se distribuye como otra variable aleatoria normal, y fijándose en el hecho de que $\mathbf{Y} \sim N(\mathbf{X}\boldsymbol{\beta}, \sigma^2 I)$ se obtiene:

$$\hat{\boldsymbol{\beta}} \sim N(\boldsymbol{\beta}, \sigma^2(\mathbf{X}'\mathbf{X})^{-1}). \tag{30}$$

Ya que $\boldsymbol{\beta}$ es un estimador lineal. Además con distribuciones marginales:

$$\hat{\boldsymbol{\beta}}_1 \sim N(\boldsymbol{\beta}_1, \sigma^2\{\mathbf{X}_1'(\mathbf{I}-\mathbf{P}_2)\mathbf{X}_1\}^{-1}). \tag{31}$$

$$\hat{\boldsymbol{\beta}}_2 \sim N(\boldsymbol{\beta}_2, \sigma^2\{\mathbf{X}_2'(\mathbf{I}-\mathbf{P}_1)\mathbf{X}_2\}^{-1}). \tag{32}$$

donde $p_1+p_2=p+1$ con p el número de variables del modelo completo.

Entonces, aplicando desde el resultado (5.1) de la lista anterior a (30) se obtiene que:

$$\sigma^{-2}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})'\mathbf{X}'\mathbf{X}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}) \sim \chi^2(k). \tag{33}$$

Y para (31) y (32):

$$\sigma^{-2}(\hat{\boldsymbol{\beta}}_1-\boldsymbol{\beta}_1)'\mathbf{X}_1'(\mathbf{I}-\mathbf{P}_2)\mathbf{X}_1(\hat{\boldsymbol{\beta}}_1-\boldsymbol{\beta}_1) \sim \chi^2(p_1).$$

$$\sigma^{-2}(\hat{\boldsymbol{\beta}}_2-\boldsymbol{\beta}_2)'\mathbf{X}_2'(\mathbf{I}-\mathbf{P}_1)\mathbf{X}_2(\hat{\boldsymbol{\beta}}_2-\boldsymbol{\beta}_2) \sim \chi^2(p_2).$$

Los residuales estimados $\hat{\boldsymbol{\varepsilon}}=\boldsymbol{Y}-\mathbf{X}\hat{\boldsymbol{\beta}}$ son una deformación del ruido $\boldsymbol{\varepsilon}$, ya que $\hat{\boldsymbol{\varepsilon}}=(\mathbf{I}-\mathbf{P})\boldsymbol{\varepsilon}$. Para ver la relación, sólo se debe notar que de la definición de $\mathbf{P}$ se verifica que $\mathbf{PX}=\mathbf{X}$ y aplicando esto a (16) se tiene que

$$\begin{aligned}
\hat{\boldsymbol{\varepsilon}} &= (\mathbf{I}-\mathbf{P})\mathbf{Y} \\
&= (\mathbf{I}-\mathbf{P})(\mathbf{X}\hat{\boldsymbol{\beta}}+\boldsymbol{\varepsilon}) \\
&= \mathbf{I}\mathbf{X}\hat{\boldsymbol{\beta}}+\mathbf{I}\boldsymbol{\varepsilon}-\mathbf{P}\mathbf{X}\hat{\boldsymbol{\beta}}-\mathbf{P}\boldsymbol{\varepsilon} \\
&= \mathbf{I}\boldsymbol{\varepsilon}-\mathbf{P}\boldsymbol{\varepsilon} \\
&= (\mathbf{I}-\mathbf{P})\boldsymbol{\varepsilon}.
\end{aligned}$$

Así, se observa que no es posible obtener la distribución muestral del ruido directamente, pues $\mathbf{I}-\mathbf{P}$ no es invertible. Pero, es posible factorizar $\mathbf{I}-\mathbf{P}$ como un el producto $\mathbf{CC}'$ con $\mathbf{C}$ una matriz de $n\times(n-p)$ de columnas ortogonales que sean ortogonales a las columnas de la matriz de diseño $\mathbf{X}$, es decir $\mathbf{C}'\mathbf{X}=\mathbf{0}$ y $\mathbf{C}'\mathbf{C}=\mathbf{I}$. Se verá como se aplica ésta factorización:

Por un lado se sabe que $\mathbf{Y}\sim N_n(\mathbf{X}\boldsymbol{\beta},\sigma^2\mathbf{I})$, lo que implica que $\mathbf{C}'\mathbf{Y}\sim N_n(\mathbf{0},\sigma^2\mathbf{I})$, o equivalentemente $\sigma^{-1}\mathbf{C}'\mathbf{Y}\sim N(\mathbf{0},\mathbf{I})$. Por lo tanto:

$$\sigma^{-2}\mathbf{Y}'\mathbf{C}\mathbf{C}'\mathbf{Y}=\sigma^{-2}(\mathbf{Y}-\mathbf{X}\hat{\boldsymbol{\beta}})'(\mathbf{Y}-\mathbf{X}\hat{\boldsymbol{\beta}})\sim\chi^2(n-p-1). \tag{34}$$

Por las proposiciones (5.1) y (5.2).

Por otro lado, $\widehat{\boldsymbol{\beta}}-\boldsymbol{\beta}\sim N(\mathbf{0}_p,\sigma^2(\mathbf{X}'\mathbf{X})^{-1})$ y del resultado de la proposición (5.2):

$$\sigma^{-2}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})'\mathbf{X}'\mathbf{X}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})\sim\chi^2(p). \tag{35}$$

Observando que $\mathbf{X}\hat{\boldsymbol{\beta}}=\mathbf{PY}$ y $\mathbf{Y}-\mathbf{X}\hat{\boldsymbol{\beta}}=(\mathbf{I}-\mathbf{P})\mathbf{Y}$ son proyecciones con una matriz de varianzas-covarianzas diagonal (esto es una implicación del hecho de que $\mathbf{P}$ es una matriz idempotente y simétrica; se comprueba observando que $\mathbf{P}'(\mathbf{I}-\mathbf{P})=\mathbf{0}_n$), y si suponemos que ambas son normales, por (34) y (35) se puede concluir que son variables independientes. Entonces:

$$F=\frac{\frac{(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})'\mathbf{X}'\mathbf{X}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})}{p}}{\frac{(\mathbf{Y}-\mathbf{X}\hat{\boldsymbol{\beta}})'(\mathbf{Y}-\mathbf{X}\hat{\boldsymbol{\beta}})}{n-p-1}}=\frac{1}{p\hat{\sigma}^2}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})'\mathbf{X}'\mathbf{X}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})\sim F(p,n-p-1). \tag{36}$$

Se procede de manera análoga (de (31)) para cualquier $\beta\in\mathbb{R}^{p+1}$ con el estadístico:

$$F=\frac{1}{p_2\hat{\sigma}^2}(\hat{\boldsymbol{\beta}}_2-\boldsymbol{\beta}_2)'\mathbf{X}_2'(\mathbf{I}-\mathbf{P}_1)\mathbf{X}_2(\hat{\boldsymbol{\beta}}_2-\boldsymbol{\beta}_2)\sim F(p_2,n-p-1). \tag{37}$$

Significancia de la regresión

Un caso particular importante de la distribución (36) sucede en $\beta = 0_p$, pues en este punto el estadístico F sirve para mostrar que al menos un parámetro del modelo tiene relación con la variable dependiente (Y).

Así, el contraste de hipótesis es el siguiente:

$$H_0 : \beta_1 = \cdots = \beta_p = 0 \quad \text{vs} \quad H_a : \exists j \text{ tal que } \beta_j \neq 0.$$

El estadístico de prueba se reduce a la forma:

$$\frac{1}{p\hat{\sigma}^2}\hat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{X}\hat{\boldsymbol{\beta}}. \tag{38}$$

ANOVA A manera de resumen se puede mostrar una tabla llamada ANOVA donde se divide la variabilidad de la variable respuesta Y de acuerdo a la fuente de la misma.

Cuadro 1: ANOVA

Fuente de Variación	Suma de cuadrados	Grados de libertad	Cuadrado medio	Estadístico F
Regresión	SCR	p	MSR	$F = \frac{MSR}{MSE}$
Residual	SCE	$n-p-1$	MSE	
Total	SCT	$n-1$		

Donde $MSR = \frac{SCR}{p}$ y $MSE = \frac{SCE}{n-p-1}$.

Algunas observaciones:

- SCE tiene $n-p-1$ grados asociados, donde $p+1$ es el número de parámetros que tienen que ser estimados.
- SCR tiene p grados de libertad, uno por cada parámetro estimado.
- SCT está asociado a $n-1$ grados de libertad, que corresponde a la suma de los grados de libertad de SCE y SCR.

Significancia de las variables regresoras

Otro caso particular relevante sucede cuando $\boldsymbol{\beta}_2 = \mathbf{0}_P$, pues se contrastan las hipótesis:

$$H_0 : \beta_{2_1} = \beta_{2_2} = \cdots = \beta_{2_{p_2}} = 0 \text{ vs } H_a : \exists j \text{ con } \beta_{2_j} \neq 0 \quad j \in \{1, \ldots, p_2\}.$$

La prueba mostrará si se puede descartar un subconjunto de variables ($\boldsymbol{\beta}_2$) del modelo de regresión o no en términos de la suma de cuadrados de dos modelos. El primero con $\boldsymbol{\beta}_2 = \mathbf{0}$, es decir, bajo la hipótesis nula y el segundo con todas las variables (H_a). Se llamará SCE_0 y SCE_a a la suma de cuadrados de los errores bajo H_0 y H_a respectivamente. Entonces el estadístico se expresa como a continuación:

$$F = \frac{SCE_0 - SCE_a \big/ p_2}{SCE_a \big/ n-p-1}. \tag{39}$$

Cuando la hipótesis afecta a solamente un parámetro β_i de $\boldsymbol{\beta}$ (es decir $\boldsymbol{\beta}_2$ es un vector de una sola entrada), es estadístico toma la siguiente forma:

$$F = \frac{(\hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_1^*)^2}{\sigma^2 w_{ii}} \sim F(1, n-p-1).$$

Donde w_{ii} es el i-ésimo elemento de la matriz $(\mathbf{X}'\mathbf{X})^{-1}$. O equivalentemente:

$$t = \frac{\hat{\boldsymbol{\beta}}_i - \hat{\boldsymbol{\beta}}_i^*}{\sqrt{\hat{\boldsymbol{\sigma}}^2 w_{ii}}} \sim T(n-p-1).$$

5.5. No multicolinealidad

El supuesto de que las variables regresoras son independientes entre si, no siempre se puede garantizar. No hay un método seguro para detectar la multicolinealidad pero existen múltiples signos para indicar su presencia. Se muestran algunos:

- Tener un R^2 alto, con los coeficientes de las pruebas t no significativos o relativamente altos.

- Coeficientes de correlación simple o parciales altos en valor absoluto.

- Estadísticos F significativos para alguna regresión auxiliar.
  Se llamará regresión auxiliar, a cada una de las regresiones que toman a la covariable X_i como variable respuesta contra los demás regresores. Para cada regresión auxiliar se debe calcular el estadístico:

  $$F_i = \frac{R^2_{x_i x_2 \ldots x_p}/p}{(1 - R^2_{x_i x_2 \ldots x_p})/(n-p-1)}.$$

  Que distribuye como una distribución $F_{p,n-p-1}$ con n el número de observaciones, $p+1$ el número de parámetros incluyendo el intercepto, $R^2_{x_i x_2 \ldots x_p}$ el coeficiente de determinación de la regresión auxiliar.

  Si F sobrepasa el valor crítico para el nivel de significancia seleccionado, no se rechazará que la variable $\mathbf{X}_i$ está correlacionada con alguno de las variables explicativas de la regresión auxiliar.

- Observando el rango de los eigenvalores de la matriz de covariables $\mathbf{X}$. Aplicado la siguente regla a los eigenvalores de $\mathbf{X}$ o equivalentemente aplicando la condición del índice.

  Sea $k = \frac{\text{máx}(eigenvalores)}{\text{mín}(eigenvalores)}$.

  Si $100 \leq k \leq 1000$ existe una correlación de moderada a grave, si k excede 1000 entonces existe una correlación severa.

  Alternativamente, si $CI = \sqrt{k}$ entonces, si $10 \leq CI \leq 30$ existe una correlación de moderada a grave, si CI excede 30 entonces existe una correlación severa.

- Observando el factor de inflación de la varianza (VIF). Se define el VIF, como:

$$VIF = \frac{1}{1 - R^2_{x_i x_2 \ldots x_p}}.$$

Donde se aplicará una regla similar a la del CI. Si $1 \leq VIF \leq 5$ la correlación se considera de nula a moderada, si $VIF > 5$ se considera grave. Una implicación importante de ésta definición es

$$Var[\widehat{\beta}_i] = \frac{\sigma^2}{\sum x_{i,j}} VIF.$$

Es decir, la varianza del estimador $\widehat{\beta}_i$ es proporcional al VIF.

Los números que se dieron como referencia para k, CI ó el VIF se encuentran en textos tan antiguos como Farrar y Glauber 1967, pero con una computadora con un sistema operativo de 64 bits Windows 8.1 Pro y un procesador Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz 3.20 GHz se pudo invertir la matriz $\mathbf{X'X}$ con una correlación de 0.99999 mediante el siguiente código:

```
#Se descarga las librerias necesarias
if (!require("MASS")) install.packages("MASS")

  #Se fija la semilla
  semilla=2175
  set.seed(semilla)

  #n es el tamano de muestra
  n = 5000

  #mu, ro y sigma son los parametros para generar los datos
  #correlacionados mediante una normal multivariada
  mu = c(0,0)
  ro = 0.99999
  sigma = matrix(c(1,ro,ro,1),2)

  #X es la matriz de datos
  X = MASS::mvrnorm(n,mu,sigma)

  #Se muestra la correlacion de los datos
  print(cor(X)[1,2], digits = 5)
  X = cbind(rep(1,dim(X)[2]),X)

  #Se invierte la matriz la matriz de disenio
  D = t(X) %*%  X
  DInv = solve(D)

  #Se comprueba que D y DInv son matrices inversas
  (D %*% DInv)
  (DInv %*% D)
```

6. Modelos lineales con regularización

Cuando se observa el estimador de mínimos cuadrados ($\widehat{\boldsymbol{\beta}} = (\mathbf{X'X})^{-1}\mathbf{X'Y}$) y su varianza ($Var[\widehat{\boldsymbol{\beta}}] = \sigma^2(\mathbf{X'X})^{-1}$) se puede ver que ambos dependen de $(\mathbf{X'X})^{-1}$. Lo que implica que sólo se puede utilizar el estimador de mínimos cuadrados cuando $(\mathbf{X'X})^{-1}$ existe, que excluye los casos donde las variables son linealmente dependientes o se tienen más variables

que observaciones (alta dimensionalidad), además que son sensibles a correlaciones en la matriz de diseño (correlaciones fuertes inflan la varianza y norma del estimador, lo que facilita el sobre-ajuste).

Una forma de atacar estos problemas es penalizar la norma de los parámetros (β_i), de ésta manera se limita la influencia que tiene cada variable en el modelo y al mismo tiempo disminuye la varianza del estimador (pues se reduce el área de búsqueda). En este sentido, se puede plantear una función de "perdida"sobre los parámetros que caracterizan al modelo.

$$L(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + R(\boldsymbol{\beta}).$$

Donde $R(\boldsymbol{\beta})$ es una función que debe penalizar la norma del vector $\boldsymbol{\beta}$ ($\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_p)'$). Comúnmente no se penaliza β_0 y se estima como una media $\widehat{\beta}_0 = \overline{Y}$ para que se pueda estimar como un plano al origen $(\mathbf{Y} - \widehat{\beta}_0) = \widehat{\beta}_1\mathbf{X}_1 + \cdots + \widehat{\beta}_p\mathbf{X}_p$. Así, sin pérdida de generalidad se supondrá:

$$\sum_{i=1}^{n} Y_i = 0$$

$$\forall j \quad \sum_{i=1}^{n} x_{i,j} = 0 \tag{40}$$

$$\forall j \quad \mathbf{X}_j'\mathbf{X}_j = 1 \tag{41}$$

Geométricamente (41) evita que el rango de cada variable X_i influya en la norma del vector (recordando $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$) y (40) garantiza que $\sum_{i=1}^{n} Y_i = \widehat{\beta}_1 \sum_{i=1}^{n} x_{i,1} + \cdots + \widehat{\beta}_p \sum_{i=1}^{n} x_{i,p} = 0$. Estas son condiciones importantes pues se usarán en cálculos posteriores.

A partir de aquí, se llamará $\widehat{\boldsymbol{\beta}}^{mc}$ al estimador de mínimos cuadrados.

6.1. Regresión Ridge

La forma más sencilla (porque es diferenciable) de aplicar regularización a los parámetros $\boldsymbol{\beta}$ es definir $R(\boldsymbol{\beta}) = \lambda \sum_{i=1}^{p} \beta_i^2$ para alguna $\lambda \geq 0$. Quedando la función de perdida como:

$$L(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|^2 \tag{42}$$

Tomando la derivada de la función de perdida respecto de $\boldsymbol{\beta}$:

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\mathbf{X}'(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) + 2\lambda\boldsymbol{\beta}.$$

Se encuentra el estimador como:

$$\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{Y}. \tag{43}$$

Ahora se verán algunas de sus propiedades.

- No es un estimador insesgado

$$\begin{aligned}
\mathbb{E}[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] &= \mathbb{E}[(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{Y}] \\
&= (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbb{E}[\mathbf{Y}] \\
&= (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{X}\boldsymbol{\beta}.
\end{aligned}$$

- $\mathbb{E}[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] \to 0$ cuando $\lambda \to \infty$

$$\lim_{\lambda\to\infty} \mathbb{E}[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] = [\lim_{\lambda\to\infty} (\mathbf{X}'\mathbf{X}+\lambda\mathbf{I})^{-1}]\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \mathbf{0}_p.$$

Las siguientes relaciones aplican cuando $(\mathbf{X}'\mathbf{X})^{-1}$ existe:

- $\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}$ está en función de $\widehat{\boldsymbol{\beta}}$.

$$\begin{aligned}
\text{Se define } \mathbf{P}_\lambda =& (\mathbf{X}'\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{X} \\
\mathbf{P}_\lambda\widehat{\boldsymbol{\beta}} &= \mathbf{P}_\lambda(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \\
&= [(\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X}+\lambda\mathbf{I})]^{-1}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \\
&= (\mathbf{X}'\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \\
&= (\mathbf{X}'\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{Y} \\
&= \widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}.
\end{aligned}$$

De lo anterior se desprende que $Var[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] = \mathbf{P}_\lambda Var[\widehat{\boldsymbol{\beta}}]\mathbf{P}_\lambda'$.

- $Var[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] \to 0$ cuando $\lambda \to \infty$.

$$\lim_{\lambda\to\infty} Var[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] = \lim_{\lambda\to\infty} \sigma^2\mathbf{P}_\lambda(\mathbf{X}'\mathbf{X})^{-1}\mathbf{P}_\lambda' = \mathbf{0}_p.$$

- $Var[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] \preceq Var[\widehat{\boldsymbol{\beta}}^{mc}]$.

$$\begin{aligned}
Var[\widehat{\boldsymbol{\beta}}^{mc}] - Var[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] &= \sigma^2[(\mathbf{X}'\mathbf{X})^{-1} - \mathbf{P}_\lambda(\mathbf{X}'\mathbf{X})^{-1}\mathbf{P}_\lambda'] \\
&= \sigma^2\mathbf{P}_\lambda\{[\mathbf{I}+\lambda(\mathbf{X}'\mathbf{X})^{-1}](\mathbf{X}'\mathbf{X})^{-1}[I+\lambda(\mathbf{X}'\mathbf{X})^{-1}] - (\mathbf{X}'\mathbf{X})^{-1}\}\mathbf{P}_\lambda' \\
&= \sigma^2\mathbf{P}_\lambda[2\lambda(\mathbf{X}'\mathbf{X})^{-2} + \lambda^2(\mathbf{X}'\mathbf{X})^{-3}]\mathbf{P}_\lambda' \\
&= \sigma^2[\mathbf{X}'\mathbf{X}+\lambda\mathbf{I}]^{-1}[2\lambda\mathbf{I}+\lambda^2(\mathbf{X}'\mathbf{X})^{-1}]\{[\mathbf{X}'\mathbf{X}+\lambda\mathbf{I}]^{-1}\}'.
\end{aligned}$$

Como la última matriz está compuesta por la multiplicación de matrices definidas no negativas, se deduce que las entradas de la diferencia entre las matrices de varianzas-covarianzas son no negativas. Por lo tanto, las entradas de la matriz de varianzas covarianzas del estimador tradicional son mayores o iguales a las del estimador de la regresión Ridge ($Var[\widehat{\boldsymbol{\beta}}_{\lambda}^{Ridge}] \preceq Var[\widehat{\boldsymbol{\beta}}^{mc}]$).

6.2. Regresión Lasso

La Regresión Lasso es un modelo de regresión lineal que minimiza la siguiente función:

$$L(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_1. \tag{44}$$

Donde $\lambda > 0$ y $\|\boldsymbol{\beta}\|_1 = \sum_{i=1}^{p} |\beta_i|$.

Lo primero que se observa es que esta función no es diferenciable, por lo que requerirá unas herramientas matemáticas distintas a las que se usaron para derivar las fórmulas de los estimadores de mínimos cuadros y Ridge. En seguida se mostrarán los resultados necesarios para comprender el método de estimación de una forma intuitiva, el lector que deseé profundizar en el tema puede revisar N. Z. Shor, 1985.

Subgradiente de funciones convexas

Proposición 6.1. *Sí $f:[a,b]\to\mathbb{R}$ es una función diferenciable, entonces:*

- *$f:[a,b]\to\mathbb{R}$ es convexa sí y sólo si $\forall\ x_0\in[a,b]\ \exists\ c(t)=f(x_0)+f'(x_0)(t-x_0)$ para $t\in[a,b]$ tal que $f(x)\geq c(x)=f(x_0)+f'(x_0)(x-x_0)$.*

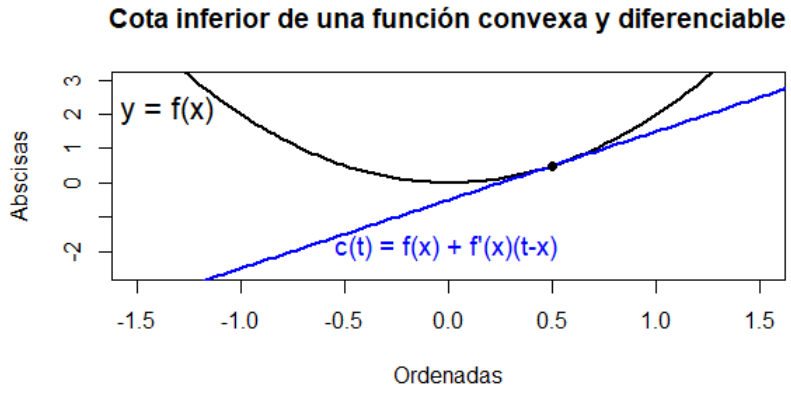

Figura 10: Gráfica de una función convexa y diferenciable ($f(x)$) y una función que la acota inferiormente ($c(t)$).

Esta última proposición expresa que para una función convexa se puede encontrar otras funciones que sean cota inferior (que puede ser importante si nuestro objetivo es encontrar un mínimo) y además menciona la forma funcional de las mismas. Esto será importante pues una de estas cotas podría indicar cuando se encuentre el punto mínimo.

Definición 6.1. *Subgradiente*
Se dice que un vector $\boldsymbol{g}\in\mathbb{R}^n$ es subgradiente de una función convexa $f:\mathbb{R}^n\to\mathbb{R}$ en $\boldsymbol{x}$ sí:

$$f(\boldsymbol{t})\geq f(\boldsymbol{x})+\boldsymbol{g}'(\boldsymbol{t}-\boldsymbol{x})$$

$\forall\ \boldsymbol{t}\in Dom_f$.

De la definición es inmediato notar que sí $\mathbf{g}=\mathbf{0}_n\in\mathbb{R}^n$ es subgradiente de $\mathbf{x}$, entonces $\mathbf{x}$ es el punto mínimo de la función pues:

$$f(\mathbf{t})\geq f(\mathbf{x})+\mathbf{0}_n'(\mathbf{t}-\mathbf{x})$$

$\forall\mathbf{t}\in Dom_f$.

Además, algunas otras propiedades son las siguientes:

- Sí f es convexa siempre existe al menos un vector subgradiente para cada punto en el dominio.
- Sí f es diferenciable entonces $g=\bigtriangledown f(\mathbf{x})$ es el único subgradiente para f en $\mathbf{x}$.

Definición 6.2. *Subdiferencial*
Al conjunto de todos los subgradientes g de una función convexa $f : \mathbb{R}^n \to \mathbb{R}$ en un punto $\boldsymbol{x}$ se le llamará subdiferencial de f en $\boldsymbol{x}$ y se denotara por $\delta f(\boldsymbol{x})$.

Por ejemplo:

Sea $|\circ| : \mathbb{R} \to \mathbb{R}^+$ la función valor absoluto. Es bien conocido que el valor absoluto no es una función difereciable, pero sus subdiferenciales permiten comprobar que el mínimo de la función se alcanza en cero.

$$|x| = \begin{cases} -x & \text{si } x < 0 \\ 0 & \text{si } x = 0 \\ x & \text{si } x > 0. \end{cases} \tag{45}$$

$$\delta|x| = \begin{cases} \{-1\} & \text{para } x < 0 \\ [-1, 1] & \text{si } x = 0 \\ \{1\} & \text{para } x > 0. \end{cases} \tag{46}$$

En la siguiente imagen se puede ver algunas de las rectas tangentes (con pendiente en $\delta|0|$) a la función en cero.

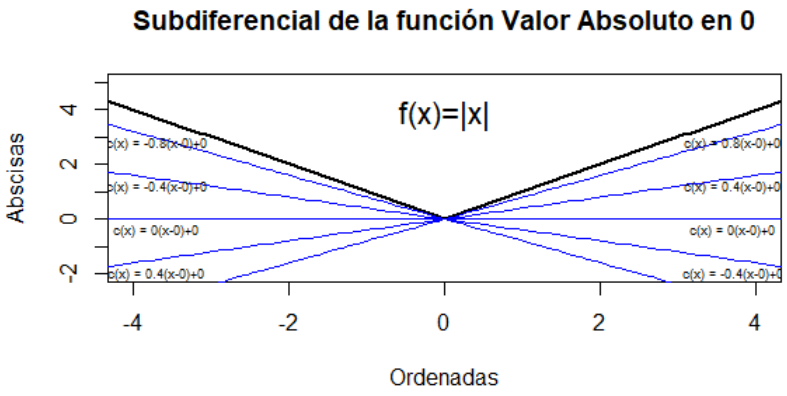

Figura 11: Gráfica de algunas rectas con pendiente en $\delta|0|$.

Una propiedad que se usará es la llamada regla de la suma de subgradientes, que permite abrir el subgradiente de una suma como suma de dos subgradientes. Queda expresada en la siguiente proposición:

Proposición 6.2. *Sean f y g funciones convexas semicontinuas inferiores, sea $x \in Dom_f \cap Dom_g$, entonces, $p \in \delta h(x)$ tal que existen $s \in \delta f(x)$ y $t \in \delta g(x)$ con $p = s + t$.*

Lasso con un sólo parámetro

Ahora se dedica el texto a encontrar el mínimo de la función de perdida $\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_1$ con las herramientas que se han visto en esta sección: Supóngase que $\boldsymbol{\beta} = \beta_1$ minimiza la función $L(\beta)$, entonces el subdiferencial $\delta L(\beta_1)$ debe contener al vector cero como subgradiente, es decir:

$$-\mathbf{X}_1'(\mathbf{Y} - \mathbf{X}_1\beta_1) + \lambda\delta|\beta_1| = 0. \tag{47}$$

De lo anterior se desprende la expresión:

$$\mathbf{X}_1'(\mathbf{Y}-\mathbf{X}_1\beta_1)=\frac{\lambda}{2}\delta|\beta_1|.$$

Como las variables están estandarizadas ($\mathbf{X}_1'\mathbf{X}_1=1$) se consigue despejar β_1 como:

$$\beta_1=\mathbf{X}_1'\mathbf{Y}-\frac{\lambda}{2}\delta|\beta_1|.$$

Analizando la última expresión por casos:

Caso: $\beta_1<0$.

$$\beta_1<0\Leftrightarrow\mathbf{X}_1'\mathbf{Y}-\frac{\lambda}{2}(-1)<0$$

$$\Leftrightarrow\mathbf{X}_1'\mathbf{Y}<-\frac{\lambda}{2}.$$

Caso: $\beta_1=0$.

$$\beta_1=0\Leftrightarrow\exists g\in[-1,1]:\mathbf{X}_1'\mathbf{Y}-\frac{\lambda}{2}g=0$$

$$\Leftrightarrow 0\in[\mathbf{X}_1'\mathbf{Y}-\frac{\lambda}{2},\mathbf{X}_1'\mathbf{Y}+\frac{\lambda}{2}]$$

$$\Leftrightarrow\quad|\mathbf{X}_1'\mathbf{Y}|\leq\frac{\lambda}{2}.$$

Caso: $\beta_1>0$.

$$\beta_1>0\Leftrightarrow\mathbf{X}_1'\mathbf{Y}-\frac{\lambda}{2}(1)>0$$

$$\Leftrightarrow\quad\mathbf{X}_1'\mathbf{Y}>\frac{\lambda}{2}.$$

En resumen

$$\widehat{\beta}_1=\begin{cases}\mathbf{X}_1'\mathbf{Y}+\frac{\lambda}{2} & \text{para } \mathbf{X}_1'\mathbf{Y}<-\frac{\lambda}{2}\\ 0 & \text{si } |\mathbf{X}_1'\mathbf{Y}|\leq\frac{\lambda}{2}\\ \mathbf{X}_1'\mathbf{Y}-\frac{\lambda}{2} & \text{para } \mathbf{X}_1'\mathbf{Y}>\frac{\lambda}{2}.\end{cases}\tag{48}$$

Para simplificar la notación se creará una función:

$$U_\lambda(x)=\begin{cases}x-\lambda & \text{para } x<-\lambda\\ 0 & \text{si } |x|\leq\lambda\\ x-\lambda & \text{para } x>\lambda.\end{cases}\tag{49}$$

Entonces $\widehat{\beta}_1=U_{\lambda/2}(\mathbf{X}_1'\mathbf{Y})$, y sumado al hecho de que $\mathbf{X}_1'\mathbf{X}_1=1$ se reescribe $\mathbf{X}_1'\mathbf{Y}$ como el estimador de mínimos cuadrados β^{mc}.Por lo tanto:

$$\widehat{\beta}_1^{lasso}=U_{\lambda/2}(\widehat{\beta}_1^{mc}).$$

Lasso con una matriz de diseño ortogonal

Cuando se tiene una matriz de diseño ortogonal ($\mathbf{X}'\mathbf{X} = \mathbf{I}$), el estimador de Lasso se obtiene siguiendo el mismo procedimiento que para una sola variable. Procediendo:

Supóngase que $\boldsymbol{\beta}$ minimiza la función de perdida. Se encontrará el vector cero dentro del subdiferencial.

$$-2\mathbf{X}'(\mathbf{Y} - \mathbf{X}_1\boldsymbol{\beta}) + \lambda \sum_{i=1}^{p} \delta|\beta_i| = 0. \tag{50}$$

De lo anterior se desprende la expresión:

$$\mathbf{X}'(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) = \frac{\lambda}{2} \sum_{i=1}^{p} \delta|\beta_i|.$$

Y por hipótesis ($\mathbf{X}'\mathbf{X} = \mathbf{I}$) se consigue despejar $\boldsymbol{\beta}$ como:

$$\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y} - \frac{\lambda}{2} \sum_{i=1}^{p} \delta|\beta_i|.$$

Y por el razonamiento que se vio en el caso de una variable

$$\widehat{\beta}_j = \begin{cases} (\mathbf{X}'\mathbf{Y})_j + \frac{\lambda}{2} & \text{para } (\mathbf{X}'\mathbf{Y})_j < -\frac{\lambda}{2} \\ 0 & \text{si } |(\mathbf{X}'\mathbf{Y})_j| \leq \frac{\lambda}{2} \\ (\mathbf{X}'\mathbf{Y})_j - \frac{\lambda}{2} & \text{para } (\mathbf{X}'\mathbf{Y})_j > \frac{\lambda}{2}. \end{cases} \tag{51}$$

Y finalmente

$$\widehat{\beta}_j^{lasso} = U_{\lambda/2}(\widehat{\beta}_j^{mc}) \quad \forall j.$$

Método de descenso por coordenadas

En un caso mas general, solamente se supondrá que la matriz $\mathbf{X}'\mathbf{X}$ es invertible, entonces se puede encontrar la siguiente relación:

$$-2\mathbf{X}_j'(\mathbf{Y} - \mathbf{X}_{-j}\beta{-}j - \beta_j\mathbf{X}_j) + \lambda \sum_{i=1}^{p} \delta|\beta_i| = 0.$$

Donde:

$\mathbf{X}_{-j}$ es la matriz compuesta de todas las columnas de $\mathbf{X}$ excepto la número j y $\boldsymbol{\beta}_{-j}$ está definido de una forma similar.

Luego de desarrollar y considerando la estandarización ($\mathbf{X}_j'\mathbf{X}_j = 1$) se obtiene una solución para cada β_j.

$$\widehat{\beta}_j = \mathbf{X}_j'(\mathbf{Y} - \mathbf{X}_{-j}\boldsymbol{\beta}_{-j}) - \frac{\lambda}{2}\delta|\beta_j|. \tag{52}$$

Se puede observar que el estimador $\widehat{\beta}_j$ está en términos de los demás parámetros ($\boldsymbol{\beta}_{-j}$), entonces no se ha conseguido una forma cerrada. Sin mencionar que el análisis en el caso donde $\mathbf{X}'\mathbf{X}$ es invertible.

En general se utiliza el siguiente algoritmo conocido como "Método del descenso por coordenadas" para estimar los parámetros:

1. Inicializar $\boldsymbol{\beta} = 0$.

2. Para $j = 1, \ldots, p$, hacer lo siguiente:

2.1. Calcular los residuales parciales.

$$\mathbf{Res}_j = \mathbf{Y} - \sum_{i \neq j} \mathbf{X}_i \beta_i.$$

2.2. Inicializar $\beta_j = \widehat{\beta}_j^{mc}$ por mínimos cuadrados.

$$\beta_j = \mathbf{Res}_j' \mathbf{X}_j.$$

2.3. Actualizar β_j con la solución de una sola variable.

$$\beta_j = U_{\lambda/2}(\widehat{\beta}_j^{mc}).$$

3. Verificar la convergencia, en caso contrario ir al segundo paso.

Interpretación geométrica

Hasta ahora sólo se ha mostrado las funciones de perdida en su forma lagrangiana, pero a continuación se reescribirán como problemas de optimización convexa (esto es posible por la Dualidad de Lagrange).

Regresión Ridge

$$\begin{aligned} \widehat{\boldsymbol{\beta}}^{ridge} &= \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \\ &\text{Sujeto a } \|\boldsymbol{\beta}\|_2^2 \leq t_1. \end{aligned}$$

Como es bien conocido la norma dos la restricción $|\boldsymbol{\beta}|_2^2 \leq t_1$ forma un área de búsqueda dentro (e incluyendo la frontera) de una circunferencia de radio t_1.

Regresión Lasso

$$\begin{aligned} \widehat{\boldsymbol{\beta}}^{lasso} &= \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \\ &\text{Sujeto a } \|\boldsymbol{\beta}\|_1 \leq t. \end{aligned}$$

Y las restricciones con la norma uno $\|\boldsymbol{\beta}\|_1 \leq t$ forman diamantes (o cuadrados rotados) al redor del origen.

Ahora, ambas restricciones limitan el espacio, pero sólo la norma uno crea esquinas en el área de búsqueda, lo que facilita que se estimen coeficientes como cero. A continuación las gráficas de las regresiones Lasso (a la izquierda) y Ridge (a la derecha). En ambas se muestra el área de búsqueda en color verde al rededor del origen y el estimador por mínimos cuadrados rodeado de elipses cuyos ejes dependen de la desviación estándar del estimador de cada β_i.

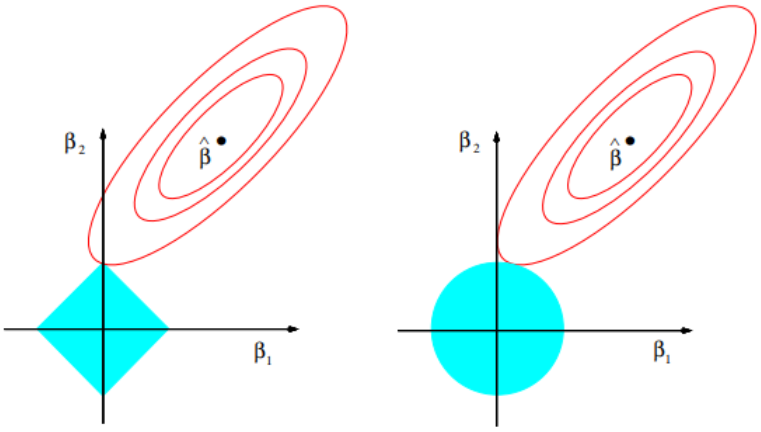

Figura 12: Intepretación gráfica de las regresiones Lasso (a la izquierda) y Ridge (a la derecha). T. Hastie, R. Tibshirani, J. Friedman. (2001). Recuperado de https://web.stanford.edu/ hastie/Papers/ESLII.pdf.

6.3. Regresión dispersa no convexa

Hasta el momento se ha visto como estimar los parámetros de un modelo de regresión lineal dadas las variables correctas, se ha planteado como calcular los parámetros de modelos con regularización (comúnmente se calcula para el modelo que contiene todas las variables) a través de funciones convexas, diferenciables y no diferenciables (con las normas $\|\mathbf{x}\|_1$ y $\|\mathbf{x}\|_2$). Ahora es de interés encontrar un método de estimación con una función de riego que penalice los parámetros con la norma cero ($\|\mathbf{x}\|_0$), que es una norma no convexa (en general normas $\|x\|_p$ con $0 \leq p < 1$ no son convexas):

$$L(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_0. \tag{53}$$

Para $\lambda > 0$. Donde la norma cero es un caso limite:

$$\|\mathbf{x}\|_0 = \lim_{q\to 0} \|\mathbf{x}\|_q = \lim_{q\to 0} (\sum_{i=1}^{p} |x_i|^q)^{\frac{1}{q}}.$$

Con:

$$\lim_{q\to 0} |x_i|^q = \begin{cases} 0 & \text{si } x_i = 0 \\ 1 & \text{si } x_i \neq 0. \end{cases}$$

Regresión dispersa no convexa con una matriz de diseño ortogonal

Como es de suponerse, la optimización de funciones no convexas es en general complejo, pero en [14] se demuestra que el punto mínimo de (53) cuando la matriz de diseño es ortogonal y estandarizado es el siguiente (se omite la demostración):

$$\widehat{\boldsymbol{\beta}}_j^{N_0} = \begin{cases} \mathbf{X}_j\mathbf{Y} & \text{si } \mathbf{X}_j\mathbf{Y} > \sqrt{\lambda} \\ 0 & \text{si } |\mathbf{X}_j\mathbf{Y}| \leq \sqrt{\lambda} \\ \mathbf{X}_j\mathbf{Y} & \text{si } \mathbf{X}_j\mathbf{Y} < -\sqrt{\lambda}. \end{cases} \tag{54}$$

Para $j = 1, \ldots, p$.

Alternativamente en [9] se crea un algoritmo que proporciona exactamente la misma solución partiendo del estimador de Lasso, llamado $Lass-0$:

Se llamará $Soporte(\boldsymbol{\beta}) = \{i | \|\beta_i\|_0 = 1\}$.

Sea $F = \{i : \|\beta_i\|_0 = 1\}$.

Sea $\mathbf{C} = (\|\beta_1\|_0, \ldots, \|\beta_n\|_0)$.

Donde $\widehat{\boldsymbol{\beta}}^{mc}$ es estimador de mínimos cuadrados para el modelo que sólo considera las variables que se encuentran marcadas con uno en el soporte, es decir:

$$\widehat{\boldsymbol{\beta}}_F^{mc} = (\mathbf{DX'XD})^{-1}\mathbf{DX'Y}.$$

Con $\mathbf{D} = diag(\mathbf{C})$.

$Lass-0:$

1. Inicializar $\boldsymbol{\beta}$ como el estimador de Lasso con la misma λ.

2. Calcular $F = Soporte(\boldsymbol{\beta})$.

3. Calcular $\boldsymbol{\beta}$ como $\widehat{\boldsymbol{\beta}}_F^{mc} = \mathbf{DX'Y}$ (recordando que $\mathbf{X'X} = \mathbf{I}$).

4. Actualizar $F = F_{\boldsymbol{\beta}}$.

5. Para cada $i \in F$ calcular $\boldsymbol{\beta}^{(i)} = \widehat{\boldsymbol{\beta}}_{F-i}^{mc}$.

6. Para cada $i \notin F$ calcular $\boldsymbol{\beta}^{(i)} = \widehat{\boldsymbol{\beta}}_{F\cup i}^{mc}$.

7. Calcular $\boldsymbol{\beta}' = \arg\min_{\boldsymbol{\beta}^{(i)}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^{(i)}\|_2^2 + \lambda\|\boldsymbol{\beta}^{(i)}\|_0$.

8. Si $\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}'\|_2^2 + \lambda\|\boldsymbol{\beta}'\|_0 < \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_0$ hacer $\boldsymbol{\beta} = \boldsymbol{\beta}'$ y regresar al punto cuatro. En caso contrario $\boldsymbol{\beta}$ es el resultado.

Proposición 6.3. *Asumiendo que $\boldsymbol{X}$ es ortogonal y estandarizado, la solución de $Lass-0$ es el punto mínimo de la función $L(\boldsymbol{\beta}) = \| \boldsymbol{Y} - \boldsymbol{X\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_0$ (51).*

Demostración por casos:

Caso $\lambda = 4$. Como $\lambda = 4$, se tiene que $\frac{\lambda}{2} = \sqrt{\lambda}$ (que son los umbrales de (51) y (54)), entonces los soportes son idénticos y ambos estimadores coinciden con los $\beta' s$ de mínimos cuadrados.

Por lo tanto $\boldsymbol{\beta}^{Lass-0} = \boldsymbol{\beta}^{N_0}$.

Caso $\lambda > 4$. Demostración por casos:

Supóngase que se puede eliminar alguna de las variables que se encuentran en la solución (51) (demostración por reducción al absurdo).

Como $4 < \lambda$ se sigue que $4\lambda < \lambda^2$ entonces $\sqrt{\lambda} < \frac{1}{2}\lambda$, es decir el umbral de

Lasso es menos restrictivo que el de (51) (manda mas variables a cero), por lo que $F_{\boldsymbol{\beta}^{N_0}} \supset F_{\boldsymbol{\beta}^{lasso}}$.
Se toma $k \in F_{\boldsymbol{\beta}^{lasso}}$ y $\boldsymbol{\beta}^* = \widehat{\boldsymbol{\beta}}^{mc}_{F-k}$ (se removió la variable número k). Para que $\boldsymbol{\beta}^*$ se aceptada como mejora tiene que ocurrir que (según el paso ocho de $Lass-0$):

$$\begin{aligned}
\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}^*\|_2^2+\lambda\|\boldsymbol{\beta}^*\|_0 &< \|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2+\lambda\|\boldsymbol{\beta}\|_0\\
\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}^*\|_2^2+\lambda(\|\boldsymbol{\beta}\|_0-1) &< \|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2+\lambda\|\boldsymbol{\beta}\|_0\\
\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}^*\|_2^2-\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2 &< \lambda.
\end{aligned}$$

Desarrollando $\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2$:

$$\begin{aligned}
\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2 &= (\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})\\
&= (\mathbf{Y}'-\boldsymbol{\beta}'\mathbf{X}')(\mathbf{Y}-\mathbf{X}\boldsymbol{\beta})\\
&= \mathbf{Y}'\mathbf{Y}-\mathbf{Y}'\mathbf{X}\boldsymbol{\beta}-\boldsymbol{\beta}'\mathbf{X}'\mathbf{Y}+\boldsymbol{\beta}\mathbf{X}'\mathbf{X}\boldsymbol{\beta}\\
&= \mathbf{Y}'\mathbf{Y}-\mathbf{Y}'\mathbf{X}\boldsymbol{\beta}-\boldsymbol{\beta}'\mathbf{X}'\mathbf{Y}+|\boldsymbol{\beta}|_2^2 \text{ por ser } \mathbf{X} \text{ ortogonal.}
\end{aligned}$$

Regresando a $\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}^*\|_2^2-\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2$.

$$\begin{aligned}
\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}^*\|_2^2-\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2 =& (\mathbf{Y}'\mathbf{Y}-\mathbf{Y}'\mathbf{X}\boldsymbol{\beta}^*-\boldsymbol{\beta}^{*\prime}\mathbf{X}'\mathbf{Y}+\|\boldsymbol{\beta}^*\|_2^2).\\
&-(\mathbf{Y}'\mathbf{Y}-\mathbf{Y}'\mathbf{X}\boldsymbol{\beta}-\boldsymbol{\beta}'\mathbf{X}'\mathbf{Y}+\|\boldsymbol{\beta}\|_2^2).\\
=& \mathbf{Y}'\mathbf{X}(\boldsymbol{\beta}-\boldsymbol{\beta}^*)+(\boldsymbol{\beta}'-\boldsymbol{\beta}^{*\prime})\mathbf{X}'\mathbf{Y}-(\|\boldsymbol{\beta}\|_2^2-\|\boldsymbol{\beta}^*\|_2^2).
\end{aligned}$$

Donde $(\boldsymbol{\beta}-\boldsymbol{\beta}^*)=(0_1,\ldots,0_{k-1},\beta_k,0_{k+1},\ldots,0_p)$, entonces $\mathbf{Y}'\mathbf{X}(\boldsymbol{\beta}-\boldsymbol{\beta}^*)=[\mathbf{Y}'\mathbf{X}(\boldsymbol{\beta}-\boldsymbol{\beta}^*)]'$.

Luego

$$\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}^*\|_2^2-\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2 = 2\beta_k(\mathbf{X}'\mathbf{Y})_k-\beta_k^2.$$

Por lo tanto

$$\begin{aligned}
2\beta_k(\mathbf{X}'\mathbf{Y})_k-\beta_k^2 &< \lambda\\
\beta_k^2 &< \lambda\\
\Rightarrow \lambda^2 &< \lambda \text{ (ya que } |(\mathbf{X}'\mathbf{Y})_k|>\lambda, \text{ pues } \beta_k\neq 0)\\
\Rightarrow \lambda &< 1!
\end{aligned}$$

Por lo tanto no es posible remover alguna variable de las que se encuentran contenidas en el soporte de (51).

Supóngase que $Lass-0$ agrega alguna de la variable.

Procediendo de la misma forma que en el caso anterior

$$\begin{aligned}
\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}^*\|_2^2-\|\mathbf{Y}-\mathbf{X}\boldsymbol{\beta}\|_2^2 &< -\lambda\\
\Rightarrow -(\mathbf{X}'\mathbf{Y})_k^2 &< -\lambda\\
\Rightarrow (\mathbf{X}'\mathbf{Y})_k^2 &> \lambda\\
\Rightarrow |(\mathbf{X}'\mathbf{Y})_k| &> \sqrt{\lambda}\\
\Rightarrow |\beta_k^{N_0}| &> 0.
\end{aligned}$$

Es decir, es posible agregar sólo las variables que se encuentren en el soporte de $\boldsymbol{\beta}^{N_0}$.

De ambos casos, se observa que si $F_{\boldsymbol{\beta}^{N_0}} \supset F_{\boldsymbol{\beta}^{lasso}}$ (Lasso tiene menos variables en su soporte que la solución de N_0) $Lass-0$ sólo puede agregar variables al soporte que estén contenidos en $F_{\boldsymbol{\beta}^{N_0}}$.

Caso $\lambda < 4$**.** Este caso se sigue de un argumento análogo al caso anterior.
Como $\lambda < 4 \Rightarrow \lambda^2 < 4\lambda \Rightarrow \lambda/2 < \sqrt{\lambda}$. Así, $F_{\boldsymbol{\beta}^{lasso}} \supset F_{\boldsymbol{\beta}^{N_0}}$, Lasso es más restrictivo que la solución en (54). Supóngase que $Lass-0$ elimina alguna variable, entonces:

$$\begin{aligned} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^*\|_2^2 - \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 &< \lambda \\ \Rightarrow \beta_k &< \lambda \\ \Rightarrow |\mathbf{X}_k\mathbf{Y}| &< \sqrt{\lambda} \\ \Rightarrow \beta_k^{N_0} &= 0. \end{aligned}$$

Se asume que $Lass-0$ agrega una variable a su soporte, entonces:

$$\begin{aligned} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^*\|_2^2 - \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 &< -\lambda \\ \Rightarrow (\mathbf{X}'\mathbf{Y})_k^2 &> \lambda \\ \Rightarrow |\mathbf{X}_k\mathbf{Y}| &> \lambda \\ \Rightarrow \beta_k^{N_0} &> 0! \end{aligned}$$

Por lo tanto, no se pueden agregar variables.

Así, aplicando reiteradamente los soportes coinciden. La interpretación de este resultado es que la estimación de Lasso es cercano (en términos de soporte) a la solución (54).

Regresión dispersa no convexa sin alta dimensionalidad

Se dedicará todo el siguiente capítulo para explicar el algoritmo de optimización probabilistica que se usará para ésta tarea.

Algoritmos genéticos

Los algoritmos genéticos son algoritmos heurísticos de optimización bio-inspirados en la evolución, de búsqueda estocástica, utilizados en optimización combinatoria donde los algoritmos tradicionales no resulten aplicables, eficientes ó como una opción a estos.

Surgen de la idea de que la evolución, desde un punto de vista matemático, es un proceso de optimización en el cual una o mas poblaciones que intercambian material genético, sometidas a una presión selectiva, van descartando a los individuos menos aptos para tener descendencia, de esta forma, los individuos con características (heredables) mas adecuadas tendrán mayor probabilidad de que su descendencia esté en las siguientes generaciones, haciendo que cambie la frecuencia de aparición de ciertos genes (entre generaciones). Así, el cambio acumulativo forma la evolución.

Se identifican como factores de evolución, la mutación, la deriva genética, la migración y la selección natural. Cada uno tiene su interpretación en los algoritmos genéticos.

7. Codificación

Cada ser vivo cuenta con alguna estrategia o mecanismos de supervivencia, de alguna forma, cada organismo representa una posible solución al problema de ¿cómo sobrevivir en el mundo natural? Imitando esta situación, los algoritmos genéticos plantean que cada punto en el espacio de búsqueda es un posible poblador. Pero, ¿cómo representar esta situación?

Como es sabido, el ADN es la molécula que dirige el desarrollo y actividades de cada célula, este material es transmitido a la descendencia mediante su duplicación. El ADN se encuentra contenido en cromosomas, cada cromosoma se divide en genes, cada gen codifica una proteína específica. Así, cada cadena de genes determina las características físicas posibles de cada individuo, es decir, para cada individuo existe una codificación única que lo representa.

Los organismos cuyos cromosomas se organizan por pares o pareados son llamados diploides, para los que no se encuentran pareados, se les da el nombre de aploides.

En este trabajo solo se consideran representaciones aploides de las posibles soluciones. Se llamará poblador a cada punto candidato a solución.

Para escoger una representación adecuada se debe considerar tanto la función a optimizar, como el contexto del problema. Aquí, se muestran algunas de ellas:

Representación binaria. La representación binaria está formada únicamente por los caracteres $0, 1$ y es, de todas las representaciones, la que codifica con la menor cantidad de símbolos. Una de sus características es que cualquier otra representación puede ser expresada en binario (con una codificación mas extensa que la original).

Representación ternaria, cuaternaria y hexadecimal. La representación ternaria, cuaternaria y hexadecimal son aquellas en donde solo se usan tres, cuatro y diez símbolos $\{0,1,2\}$, $\{0,1,2,3\}$, $\{0,...,9,A,B,C,D,E,F\}$ respectivamente. Estas representaciones son mas compactas que la binaria y representa variables categóricas u ordinales, con a lo mas tres, cuatro o diez elementos. Claramente, son casos particulares de representaciones con $n \in N$ símbolos.

Algo interesante de notar es que el ADN tiene una representación cuaternaria, pues se compone solo de cuatro proteínas $\{Adenina, Citosina, Guanina, Timina\}$.

Representación entera. Esta representación usa los símbolos de los números naturales para expresar problemas con variables ordinales o categóricas en un rango finito.

Representación real. La representación real usa números reales como alfabeto, se usa para la optimización en dominios continuos. En la práctica, se limita el número de decimales a una precisión suficiente para el investigador.

Representación mixta ó híbrida. Cuando el domino de la función objetivo está compuesto por mas de un tipo de variables (categóricas, ordinales, continuas), se puede considerar una representación donde cada gen tenga su propio alfabeto. A esta codificación se llamará mixta o híbrida.

8. Población

> "Las poblaciones, no los individuos, son las unidades de evolución"
>
> (Barbadilla A. 2010).

Se entenderá por población, a un grupo de organismos que habitan en una misma región geográfica, al mismo tiempo y con la capacidad de intercambiar material genético. Se llamará genoma al conjunto de todos los genes de una población. Considerando que la carga genética de cada individuo permanece constante durante toda su vida, la variabilidad genética sólo puede surgir a partir de una población.

Los algoritmos genéticos usan matrices $\mathbf{M}_{r,s}$, con entradas en el alfabeto apropiado, como representación del genoma de una población. Donde r es el número de organismos y s el número de genes de cada uno de ellos. Comúnmente se inicializa cada matriz $M_{r,s}$ de forma aleatoria, donde se usa una distribución uniforme con soporte discreto para alfabetos discretos y sobre algún rango definido por el investigador en caso de continuos. O se podría inicializar de manera distinta si se cuenta con información adicional sobre la localización de los puntos óptimos.

Existen múltiples algoritmos genéticos que funcionan con mas de una población a la vez, teniendo una relación importante con el cómputo distribuido, pero en este trabajo sólo se escribe para una población.

9. Operadores genéticos

9.1. Operador de mutación

Una mutación es un cambio aleatorio en la secuencia de ADN de un individuo y es la principal fuente de variabilidad en el genoma de la especie.

Una alta tasa de mutación aumenta la probabilidad de adaptación de una especie a cambios ambientales a través de generar individuos con oportunidad de adaptarse a los mismos, pues a mayor tasa, mayor exploración de variantes genéticas. Pero al mismo tiempo, la mayoría de las mutaciones son desfavorables para los individuos que las porten pues pierden adaptación a las condiciones actuales. Cada especie y gen posee su propia tasa de mutación, adaptada a sus necesidades.

El operador de mutación de los algoritmos genéticos imita el proceso biológico de mutación a través de cambios aletarios en la codificación del individuo reemplazando los símbolos de su representación por otros dentro del mismo alfabeto. Para lograr lo anterior, se recorre cada gen (i) y se cambia con probabilidad p_i. Comúnmente se usa la misma tasa en todos los genes e individuos.

Para alfabetos finitos. Siempre que el alfabeto sea finito (casos particulares de representaciones enteras como la binaria, ternaria, cuaternaria ó hexadecimal) el cambio aleatorio de símbolo puede provenir de una distribución uniforme discreta, teniendo la ventaja de que cada símbolo aparece con la misma frecuencia. En la práctica, los alfabetos siempre son finitos.

Para alfabetos subconjuntos de $\mathbb{R}$. Si se tiene un alfabeto muy extenso, en muchas ocasiones es conveniente tratarlo como si estuviera compuesto por números reales (suponiendo que se puede asociar a elementos de $\mathbb{R}$), por lo que es útil establecer diferentes estrategias de mutación. Lo mas natural es trabajar con un operador de mutación que proponga modificaciones a través de simular alguna variable aleatoria continua cuyo soporte cubra el rango del alfabeto. La mutación mas tradicional en este tipo de situaciones es sumar un número generado de una variable aleatoria normal $N(0,\sigma^2)$ al gen a mutar (para alguna σ^2 propuesta).

Para representaciones mixtas. Se dirá que una representación es mixta, si genes distintos tienen diferentes alfabetos. Así, el operador de mutación debe estar adaptado al alfabeto de cada gen para poder cubrir todo tipo de representaciones.

A continuación se muestra el pseudo-código del operador de mutación tradicional:

1. Determinar el largo de la cadena $(C_1C_2\ldots C_s)$ del cromosoma (esto se define desde la representación).

2. Definir $p \in (0,1)$ la probabilidad de mutación del gen.

3. Definir $\sigma^2 > 0$ si se usan genes con representación continua, y contar m el número de elementos en el alfabeto en caso de que se usen genes discretos.

4. Desde $i = 1$ hasta s repetir los pasos del 5 al 7.

5. Generar un número aleatorio X de una variable aleatoria $Bin(1, p)$.

6. Si $X = 0$ saltar al número 7, en caso contrario:

6.1. Si el gen es discreto aplicará lo siguiente:

6.1.1. Generar un número aleatorio Y de una variable aleatoria discreta $Unif(1, m)$.

6.1.2. Asignar a C_i el símbolo número Y (ó equivalentemente Y).

6.2. En caso contrario:

6.2.1. Generar un número aleatorio Z de una variable aleatoria $N(0, \sigma^2)$.

6.2.2. Asignar a C_i el número $C_i + Z$.

7. Aumentar en 1 el valor de i.

9.2. Operador de cruzamiento

La deriva genética es el proceso estocástico que sortea material genético de la descendencia de una población, teniendo una fluctuación aleatoria en las frecuencias de los alelos de una población a sus descendientes, donde solo es posible transmitir los alelos presentes en los padres. Si en el proceso se pierde un alelo, este no vuelve a aparecer. De esta forma, la deriva genética actúa como una fuerza que disminuye la variabilidad del genoma, en este sentido es un proceso opuesto a la mutación.

John Holland integra la deriva genética a los algoritmos genéticos como un operador que llamo de cruzamiento por punto, donde se sortea un número $j \in \{2, \ldots, s-1\}$ de una distribución uniforme discreta, que parte el material genético de cada padre en dos bloques, el primero ($\mathbf{M}_{i,1:j}$) que contiene los genes que van desde 1 hasta j, el siguiente bloque contiene los restantes($\mathbf{M}_{i,(j+1):s}$) del padre i. De combinar los bloques $\mathbf{M}_{i,1:j}$ con $\mathbf{M}_{k,(j+1):s}$ y $\mathbf{M}_{k,1:j}$ con $\mathbf{M}_{i,(j+1):s}$ de los pobladores i y k respectivamente se producen dos cadenas, que se nombrarán como "hijos".

Dadas las cadenas $(H_{1,1}H_{1,2}\ldots H_{1,s})$ y $(H_{2,1}H_{2,2}\ldots H_{2,s})$ el operador de cruzamiento generaría otras dos cadenas de acuerdo al siguiente pseudo-código:

Sin pérdida de generalidad, se cruzarán los pobladores $i = 1, 2$.

Se nombrará $\mathbf{H}_h = (H_{h,1}H_{h,2}\ldots H_{h,s})$ con $h \in \{1, 2\}$ a las cadenas resultantes del proceso, ambas de tamaño s.

1. Determinar el largo de las cadenas s.

2. Generar un número aleatorio j de una distribución $Unif(2, s-1)$ discreta.

3. Desde $h = 1$ hasta 2 repetir los pasos 3.1 y 3.2.

3.1. Desde $k = 1$ hasta s repetir los pasos del 3.1.1 al 3.1.3.

3.1.1. Sea una variable entero p tal que, si $k \leq j$ hacer $p = h$ en otro caso $p = 3 - h$.

3.1.2. Hacer $H_{h,k} = C_{p,k}$.

3.1.3. Hacer $k = k + 1$.

3.2. Hacer $h = h + 1$.

9.3. Operador de selección

> "A esta conservación de las variaciones y diferencias individuales favorables y a la destrucción de las que son perjudiciales, la he llamado selección natural o supervivencia de los más aptos"
>
> (Charles Darwin, 1859).

La selección natural es un proceso de reproducción que limita la tasa de reproducción de variantes alélicas y favorece las que mejoren la adaptación de la especie y requiere de los siguientes tres factores:

1. Variación fenotípica. Para una característica de la especie, debe existir diferencias entre los individuos de la especie (es decir, debe haber variedad).

2. Reproducción diferencial. Se dice que la reproducción de un organismo es diferencial cuando se produce por causa de al menos una diferencia que esté presente comparado con otros organismos de su especie, es decir, debe existir una relación entre la adaptación de los individuos y la probabilidad de reproducción.

3. Herencia de la variación. Debe existir una relación entre el material genético y el fenotipo (características observables) del organismo.

Así, el efecto de estos factores a lo largo de las generaciones produce la evolución (es decir, se trabaja de manera acumulativa).

Traduciendo los elementos de la selección natural a un lenguaje en términos mas matemáticos:

1. La variación fenotípica indica que F_{Obj} no es una función constante.

2. La reproducción diferencial se traduciría como la existencia de un método de discriminación donde se seleccionen los pobladores mejor adaptados (auxiliándose de la función objetivo) con mayor probabilidad para aplicarles el operador de cruzamiento.

3. Y la herencia de la variación se puede interpretar como la existencia de una dependencia entre $F_{Obj}(x)$ y su descendencia.

Los puntos 1 y 3 hablan de la función a optimizar y se considera que está dada por el propio problema. Centrándose en 2, los algoritmos genéticos llaman "Operador de selección" al método que permite hacer la discriminación de los pobladores que se reproducirán de los que no. También hay que mencionar que el operador de selección depende de si el problema es multiobjetivo o no (mono-objetivo), es decir si $q = 1$ ó $q > 1$.

Para los problemas mono-objetivos las propuestas de operador de selección mas comunes son:

Selección por ruleta o proporcional

Este operador asigna probabilidades de selección proporcionales a la adaptación del individuo. El individuo mas apto será el que tendrá mayor probabilidad de ser seleccionado, en este caso el problema de optimización se observa como uno de maximización y se requiere que F_{Obj} sea una función positiva.

El operador queda como a continuación:

1. Calcular la suma de los fitness.
$$S_f = \sum_{i=1}^{M} f_i.$$
2. Para el primer poblador se define el rango R_1 como:
$$R_1 = (R_{1,1}, R_{1,2}) = (0, \frac{f_1}{S_f}).$$
3. Para cada $i = 2, \ldots, M$ se define el rango R_i como:
$$R_i = [R_{i,1}, R_{i,2}) = [R_{i-1,2}, R_{i-1,2} + \frac{f_i}{S_f}).$$
4. Generar un número aleatorio U de una distribución $Unif(0,1)$.
5. Se seleccionará al poblador x_j tal que $U \in R_j$.

La selección por ruleta es, desde un punto el punto de vista del muestreo estadístico, el mismo procedimiento que se usa en la obtención de una muestra aleatoria con probabilidades de selección proporcional al tamaño, y el poblador elegido es aquel asociado a la última estadística de orden. Cochran, W. G. (1977) es un libro de referencia para el lector interesado en muestreo.

Selección por torneo

La selección por torneo empieza por tomar un subconjunto aleatorio, de una distribución uniforme sobre la población, de tamaño T toma un subconjunto de la población y a partir de éste, seleccionar al poblador con mejor adaptación. Así, para aplicar este operador se necesita que el usuario defina un número T con $T \in \{2, \ldots, r\}$. Luego, el operador queda como a continuación:

1. Para cada $i = 1, \ldots, r$ generar un número aleatorio U_i de una distribución $Unif(0,1)$.
2. Calcular las estadísticas de orden $U_{(1)}, \ldots, U_{(T)}$.
3. Seleccionar el poblador X_i tal que:
$$f_i = min\{f_j | U_j \in \{U_{(1)}, \ldots, U_{(T)}\}\}.$$

Este operador es análogo al procedimiento que se realiza para tomar una muestra aleatoria simple de una población para luego elegir el poblador que está relacionado a la primera estadística de orden pues para este caso se ve el problema de optimización como uno de minimización.

Selección por ranking

La selección por ranking es una selección proporcional a una función que se llamará *Ranking* que es una transformación de la función objetivo, la posición que toma el poblador en un ordenamiento (de acuerdo a la función objetivo) y un parámetro P_s que se nombrará por "presión de selección". Procediendo:

$$Ranking(i) = 2 - P_s + 2(P_s - 1)\frac{j-1}{r-1}.$$

Donde:

- El poblador i posee el valor objetivo f_i que es la j-ésima estadística de orden, es decir $f_i = f_{(j)}$.
- $1 \leq P_s \leq 2$.

Así el pseudo-código es el siguiente:

1. Calcule las estadísticas de orden $f_{(1)}, \ldots, f_{(r)}$.
2. A cada poblador X_i se le asociará el número $Rank_i = Ranking(i)$.
3. Para el primer poblador se define el rango R_1 como:

$$R_1 = [R_{1,1}, R_{1,2}) = [0, Rank_1).$$

4. Para cada $i = 2, \ldots, r$ se define el rango R_i como:

$$R_i = [R_{i,1}, R_{i,2}) = [R_{i-1,2}, R_{i-1,2} + Rank_i).$$

5. Generar un número aleatorio U de una distribución $Unif(0, R_{r,2})$.
6. Se seleccionara al poblador x_j tal que $U \in R_j$.

Una de las características de este tipo de selección es que controla la esperanza del número de ocasiones que se elige al poblador mejor adaptado, pues ésta esperanza es el parámetro de selección.

Cuando se habla de un algoritmo genético multiobjetivo, las estrategias de selección son algo mas complejas por el echo de que no se pueden ordenar los elementos de $\mathbb{R}^q$, a continuación se presentan brevemente algunas tácticas exitosas para crear operadores de selección para $q > 1$:

1. Basadas en agregación: Este grupo ataca el problema a través de convertir una función multiobjetivo a una mono-objetivo. Por ejemplo, a partir de la suma ponderada de cada uno de las entradas del vector de valores $F_{Obj}(x)$.
2. Basadas en la población: En esta clase de operadores de selección se usa la táctica de crear subpoblaciones por la selección de ruleta aplicada a cada una de las componentes del vector $F_{Obj}(x)$ y luego se cruzan las mismas, así garantizan que los mejores pobladores de cada entrada de la función fitness genere descendencia (con cierta probabilidad).

3. Basadas en el concepto de optimalidad de pareto: En este caso la selección se realiza con la idea de que un poblador x_1 es preferible a uno x_2 si y sólo si todas las entradas del vector $F_{Obj}(x_1)$ son más (al menos una) o igual de optimas que las de $F_{Obj}(x_2)$ y cuando esto no se cumple, se prefiere al individuo que aporte mayor diversidad.

   Posteriormente se describirán conceptos importantes para los algoritmos multiobjetivo, la persona que deseé más información puede dirigirse a Chaman I. C., 2012.

10. Vista de alto nivel de un algoritmo genético

En esta sección se articulan los elementos de los algoritmos genéticos, el pseudo-código es el siguiente:

1. Inicialización de la población:

   En esta fase se genera la población inicial, generalmente de manera aleatoria.

   - Para una representación binaria es habitual inicializar el genoma con entradas simuladas de una distribución Bernoulli de parámetro 0.5.
   - Con un alfabeto entero, se hace un sorteo de sus elementos donde la probabilidad de aparición es homogénea.
   - Y para alfabetos subconjuntos de los reales, lo más frecuente es formar la matriz de población a partir de entradas simuladas de una distribución uniforme continua sobre un rango propuesto por el investigador.

2. Calcular la función objetivo para cada poblador.

3. Evaluar si se ha cumplido la condición de paro.

   Las condiciones de paro pueden ser diversas, entre las mas comunes se encuentran:

   - Parar el algoritmo cuando exista un poblador cuyo fitness sobrepase el mínimo previamente definido por el usuario.
   - Terminar la ejecución cuando se cumpla un tiempo de cálculo o un número de generaciones.
   - Finalizar el algoritmo si las mejoras entre generaciones no son relevantes.
   - Alguna combinación de las anteriores.

4. Crear nuevos cromosomas a partir de la población actual:

   4.1. Para cada poblador: aplicar el operador se selección para escoger una pareja.
   4.2. Para cada pareja: aplicar el operador de cruce.
   4.3. Para los descendientes de cada pareja: aplicar el operador de mutación.

5. Seleccionar los supervivientes y regresar al paso 3.

   Para mantener fijo el tamaño de la población se debe seleccionar los individuos que formaran la próxima generación, los criterios mas comunes son:

   Por fitness Sólo los mejores pobladores sobreviven (padres e hijos).
   Por edad Sólo los mejores desentiendes sobreviven.

11. Algoritmos genéticos multiobjetivo

Intuitivamente un algoritmo genético se considera multiobjetivo si está preparado para optimizar una función $F_{Obj}(\mathbf{x}) : \mathbb{R}^p \to \mathbb{R}^q$ para $q > 1$, es decir, cuenta con la capacidad de minimizar mas de una función simultáneamente (que corresponden a las entradas del vector $F_{Obj}(\mathbf{x}) = (F_1(\mathbf{x}), ..., F_q(\mathbf{x}))$). Para aclarar la idea de óptimo en problemas multiobjetivo se introduce la definición de dominancia.

Definición 11.1. *Dominancia de Pareto*

Se dice que $\boldsymbol{x}_1$ *domina a* $\boldsymbol{x}_2$ *(*$\boldsymbol{x}_1 \prec \boldsymbol{x}_2$*) si:*

Para toda $i = 1, ...q$ *se cumple* $F_i(\boldsymbol{x}_1) \leq F_i(\boldsymbol{x}_2)$ *y existe* i_0 *tal que* $F_{i_0}(\boldsymbol{x}_1) < F_{i_0}(\boldsymbol{x}_2)$.

Desde el punto de vista de la Dominancia de Pareto, una solución $\mathbf{x}$ es optima si no es dominada por algún otro elemento del espacio de búsqueda, a este conjunto se llamará Frontera de Pareto.

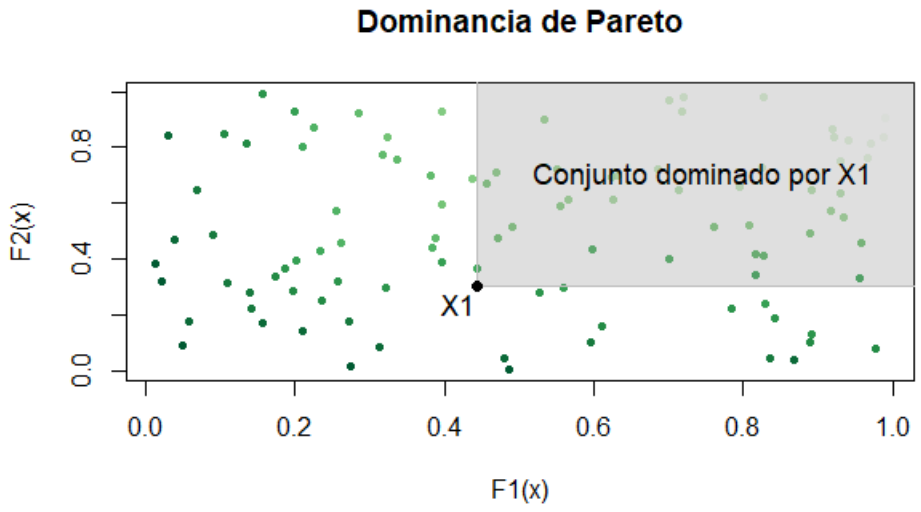

Figura 13: Gráfica donde se muestra el conjunto (constituido por el área y frontera de color gris) dominado por el punto x_1.

12. NSGA-II

El algoritmo NSGA-II (Nondominated Sorting Genetic Algorithm II) es un algoritmo genético multiobjetivo de optimización continua que usa los siguientes procedimientos:

12.1. Ordenamiento rápido no dominado

Del procedimiento que se presenta a continuación se deduce un tipo de orden para los elementos de una población a partir de particionar la misma en clases (que se llamarán fronteras), donde la primera (F_1) contiene todos los elementos que ningún poblador puede

dominar (dentro de la misma muestra de pobladores), los elementos de la segunda clase (F_2) sólo pueden ser dominados por elementos en F_1, y los de F_3 sólo por elementos en F_1 ó F_2 y así sucesivamente.

1. Para cada $p \in M$.

1.1. Sea $S_p = \emptyset$ (conjunto de elementos dominados por p).

1.2. Sea $n_p = 0$ (número de elementos que dominan a p).

1.3. Para cada $q \in M$:

1.3.1. Si $p \prec q$ (p domina q):
$S_p = S_p \cup \{q\}$.
Si $p \succ q$ (q domina p):
$n_p = n_p + 1$.

1.4. Si $n_p = 0$:
$p_{rank} = 1$ (rango del elemento p).
$F_1 = F_1 \cup \{p\}$ (se define la primera frontera de Pareto).

2. Sea $i = 1$ un contador que corre sobre el número de fronteras.

3. Mientras $F_i \neq \emptyset$:

3.1. Sea $Q = \emptyset$.

3.2. Para cada $p \in F_i$:

3.2.1. Para cada $q \in S_p$:

3.2.1.1. $n_q = n_q - 1$.

3.2.1.2. Si $n_q = 0$:
$q_{rank} = i + 1$.
$Q = Q \cup \{q\}$.

3.3. Hacer $i = i + 1$.

3.4. Hacer $F_i = Q$.

12.2. Crowding distance

Si se quisiera seleccionar el subconjunto de las mejores soluciones (con alguna cardinalidad) dentro de una frontera F_i, ya no se podría tomar la elección de acuerdo al criterio de dominancia, pues ninguno de sus elementos puede dominar a otro dentro de la misma frontera F_i (por construcción); se debe tomar algún criterio auxiliar. En este sentido la "Crowding Distance" es una herramienta que permite ordenar los elementos de mayor a menor para poder tomar los primeros mejores. Se calcula de acuerdo al siguiente método.

1. Sea $l = |\mathbf{I}|$ ($|\mathbf{I}|$ es la cardinalidad de $\mathbf{I}$).

2. Para cada $i \in \mathbf{I}$ hacer $\mathbf{I}[i]_{distance} = 0$ (inicializar la distancia como 0).

3. Para una de las entradas (m) de los vectores de la función objetivo:

1.1. $\mathbf{I} = sort(\mathbf{I}, m)$ (ordenar $\mathbf{I}$ de acuerdo a la entrada m).

1.2. $\mathbf{I}[1]_{distance} = \mathbf{I}[l]_{distance} = \infty$ (se asegura que los valores extremos se encuentren en los primeros lugares).

1.3. Para $i = 2$ hasta $(l-1)$:

1.3.1. $\mathbf{I}[i]_{distance} = \mathbf{I}[i]_{distance} + (\mathbf{I}[i+1]_{distance,m} - \mathbf{I}[i-1]_{distance,m})/(f_m^{max} - f_m^{min})$.

Donde $\mathbf{I}[i]_{distance,m}$ es el valor de la función objetivo en la entrada m para el poblador i.

12.3. Operador Crowded-Comparison

El operador Crowded-Comparison $\prec_n$ es una función que dados dos pobladores, determina cual es la "mejor" solución de acuerdo a los siguientes atributos:

1. Rango i_{rank} (se calcula en el ordenamiento rápido no dominado).

2. Crowding distance $i_{distance}$.

Definición 12.1. *Operador Crowded-Comparison*

Se dice que $i \prec_n j$ *si:*
$i_{rank} < j_{rank}$ *ó* $((i_{rank} = j_{rank})$ *y* $(i_{distance} = j_{distance}))$.

12.4. Operador de selección por Crowded-Comparison

El operador de selección que usa el NSGA-II utiliza el proceso de torneo de M individuos apoyado por el operador Crowded-Comparison para elegir el mejor poblador.

12.5. Operador de cruzamiento (SBX)

NSGA-II usa un operador de cruzamiento llamado "Simulated Binary Crossover (SBX)"

1. Sea u_i un número generado de una distribución $Unif(0,1)$, para $i = 1,2$.

2. Para $i = 1,2$ calcular:

$$\beta_i = \begin{cases} (2u)^{\frac{1}{\eta_c+1}} & \text{Si } u_i \leq 1/2 \\ [2(1-u_i)]^{-\frac{1}{\eta_c+1}} & \text{en otro caso.} \end{cases}$$

3. Calcular:

$$\mathbf{X}_1^{t+1} = (1/2)[(1+\beta_1)\mathbf{X}_1^t + (1-\beta_1)\mathbf{X}_2^t].$$

$$\mathbf{X}_2^{t+1} = (1/2)[(1-\beta_2)\mathbf{X}_1^t + (1+\beta_2)\mathbf{X}_2^t].$$

Donde:

- η_c es un número no negativo.
- $\mathbf{X}_1^t, \mathbf{X}_2^t$ son las soluciones de la generación t.
- $\mathbf{X}_1^{t+1}, \mathbf{X}_2^{t+1}$ es la descendencia.

12.6. Operador de mutación polinomial

El operador de mutación polinomial funciona de la siguiente manera:

1. Se simula una observación $mutar \sim Ber(p)$.

Si $mutar = 0$ entonces regresar $\mathbf{Y}^{t+1} = \mathbf{X}^{t+1}$ en caso contrario:

1.1. Se simula observación $r \sim Unif(0,1)$.

1.2. Calcular:

$$\delta = \begin{cases} (2r)^{\frac{1}{\eta_m+1}} & \text{Si } r \leq 1/2 \\ [1-2(1-u_i)]^{\frac{1}{\eta_m+1}} & \text{en otro caso.} \end{cases}$$

1.3. Arrojar como salida: $\mathbf{Y}^{t+1} = \mathbf{X}^{t+1} + (\mathbf{X}_U^{t+1} - \mathbf{X}_L^{t+1})\delta$.

Donde:

- η_m es un número no negativo.
- $\mathbf{X}^t$ es la solución a mutar.
- $\mathbf{X}_U$ es un vector con los límites superiores de cada variable de decisión.
- $\mathbf{X}_L$ es un vector con los límites inferiores de cada variable de decisión.
- $\mathbf{Y}^{t+1}$ es la solución después de aplicar el operador de mutación.

12.7. NSGA-II (vista de alto nivel)

Como se dijo anteriormente, NSGA-II es un algoritmo genético multiobjetivo y ahora se mostrará como interactuan los procedimientos que se acaban de mostrar:

1. Generar la población inicial M (de tamaño r y posiblemente generada de una distribución uniforme sobre el espacio de búsqueda).

2. Evaluar cada elemento en M.

3. Ranquear M (usando el "Ordenamiento rápido no dominado" y la "CrowdingDistance").

4. Seleccionar los r mejores pobladores (con r el tamaño de la población definido por el usuario, auxiliándose del operador Crowded-Comparison $[\prec_n]$).

5. Evaluar la condición de paro, si no se cumple:

- 5.1. Elegir r individuos con el Operador de Selección por Crowded-Comparison.
- 5.2. Aplicar el Operador de Cruzamiento (SBX) para crear una segunda población (Q).
- 5.3. Aplicar el Operador de Mutación Polinomial (a los elementos de Q).
- 5.4. Combinar la nueva población con la original ($M = M \cup Q$).
- 5.5. Continuar a partir del punto 2.

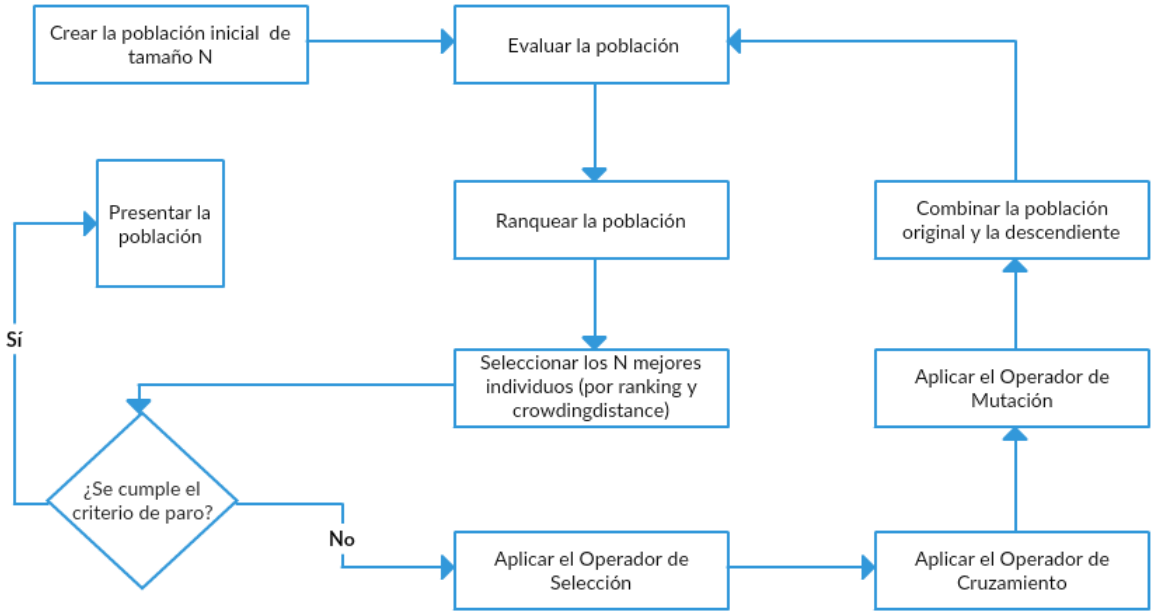

Figura 14: Diagrama de flujo del algoritmo NSGA-II.

13. NSGA-II adaptado a regresión lineal

13.1. Codificación binaria para regresión lineal

Para poder aplicar un algoritmo genético en la búsqueda de un o algunos modelos de regresión lineal (para cualquiera que sea la función objetivo) es necesario tener una codificación que represente a los mismos, pues a partir de ésta se definirá el operador de mutación y se definirá el método para inicializar la población.

El modelo de regresión lineal estimado descompone cada observación de la variable dependiente $\mathbf{Y}$ en una combinación lineal de las covariables $\mathbf{X}_1, \ldots, \mathbf{X}_p$:

$$\widehat{Y}_i = \sum_{j=0}^{p} \widehat{\beta_j} x_{ij}.$$

Con $x_{ij} = 1$ para $j = 0$.

La teoría de regresión lineal propone un método para poder llegar a los estimadores $\widehat{\beta_j}$ partiendo de los datos y muestra sus propiedades, particularmente que son estimadores consistentes. Es decir, si se pudiera incrementar el número de observaciones tanto como se quisiera se tendría una aproximación al modelo real como se desee. Donde $\widehat{\beta_j} \overset{c.s.}{\to} 0$ si la variables $\mathbf{X}_j$ no se encuentra en el modelo que crea la variable $\mathbf{Y}$.

Pero es bien conocido que introducir todas las variables dentro del modelo no siempre es lo mejor, pues incrementa la varianza de la estimación. Por lo que surge la necesidad de tener un método para poder seleccionar un subconjunto adecuado de variables (que es equivalente a suponer que $\beta_j = 0$ para algunas j). Así, cada modelo puede ser codificado como un vector $\mathbf{v} \in \{0,1\}^p$ donde:

0. indica que $\widehat{\beta}_j = 0$.

1. indica que $\widehat{\beta}_j$ es el estimador de mínimos cuadrados.

Es importante mencionar que para decodificar un vector binario a un modelo de regresión, lo primero es fijar las $\widehat{\beta}_j = 0$ y después calcular el estimador de mínimos cuadrados de cada parámetro, pues esto es equivalente a eliminar las columnas j de la matriz de diseño.

13.2. Función objetivo

Como se ha dicho al final del capítulo anterior, nuestro objetivo es proponer un algoritmo capaz de encontrar el punto mínimo de la función de perdida (53) de un modelo de regresión lineal que penalice los parámetros con la norma cero. Se reescribe la función para tenerla presente:

$$L(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_0.$$

Con $\lambda > 0$.

De observar lo anterior es intuitivo proponer la función objetivo como:

$$\begin{aligned} &R_G : \{0,1\}^p \longrightarrow \mathbb{N} \times \mathbb{R}^+ \\ &R_G(\mathbf{V}) = (\|\widehat{\boldsymbol{\beta}}^{mc}\|_0, \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}^{mc}\|_2^2). \end{aligned} \tag{55}$$

Donde $\widehat{\boldsymbol{\beta}}^{mc}$ es el estimador de mínimos cuadrados del modelo que sólo considera las variables que se encuentran marcadas como 1 en $\mathbf{V}$, es decir, $\mathbf{X}_i$ se encuentra en el modelo sí $v_i = 1$.

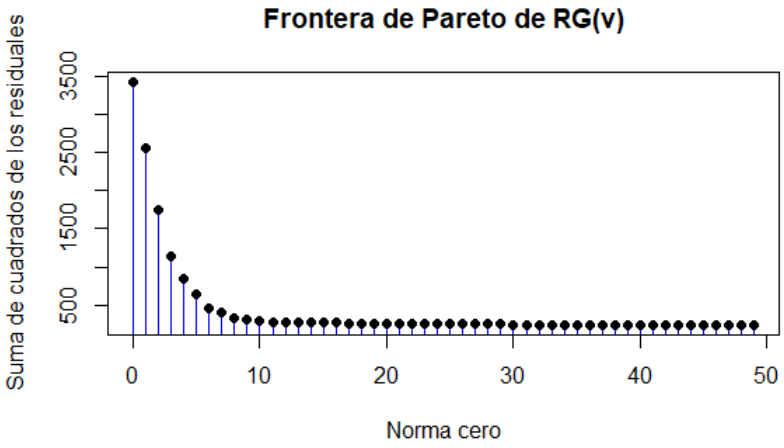

Figura 15: Gráfica de una típica Frontera de Pareto de la función descrita en (55) calculada mediante el algoritmo NSGA-II adaptado.

Proposición 13.1. *$\widehat{\boldsymbol{\beta}}^{N_0}$ minimiza (53) entonces existe un vector $\boldsymbol{v}$ en la frontera de Pareto de R_G tal que si $\widehat{\boldsymbol{\beta}}^{mc}$ es la estimación de la regresión con soporte $\boldsymbol{v}$, entonces $\widehat{\boldsymbol{\beta}}^{mc} = \widehat{\boldsymbol{\beta}}^{N_0}$.*

Demostración:

Primero se mostrará que $\widehat{\boldsymbol{\beta}}^{N_0}$ tiene que ser el estimador de mínimos cuadrados:

Sea $\lambda > 0$.
Sea $S = Soporte(\widehat{\boldsymbol{\beta}}^{N_0})$ y $\widehat{\boldsymbol{\beta}}_S^{mc}$ el estimador de mínimos cuadrados con las variables del soporte de $\widehat{\boldsymbol{\beta}}^{N_0}$. Como la norma dos es convexa se tiene:

$$\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_S^{mc}\|_2^2 \leq \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}^{N_0}\|_2^2 \quad \text{(ya que } \widehat{\boldsymbol{\beta}}_S^{mc} \text{ es el punto mínimo para la norma dos)}$$
$$\|\widehat{\boldsymbol{\beta}}_S^{mc}\|_0 \leq \|\widehat{\boldsymbol{\beta}}^{N_0}\|_0 \quad \text{(por construcción).}$$

Y se observa que se tiene que dar la igualdad en $\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_S^{mc}\|_2^2 \leq \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}^{N_0}\|_2^2$, de lo contrario $\widehat{\boldsymbol{\beta}}^{N_0}$ no sería punto mínimo. Y como la norma dos es convexa (tiene un único punto mínimo) se concluye que:

$$\widehat{\boldsymbol{\beta}}^{N_0} = \widehat{\boldsymbol{\beta}}_S^{mc}.$$

En lo que resta de la demostración sólo se escribirá $\boldsymbol{\beta}$ para referirse al punto mínimo de la ecuación (53).

El paso que sigue es mostrar que las soluciones de (53) para cualquier $\lambda > 0$ se encuentran en la frontera de Pareto, y que cualquier punto en la frontera de Pareto es solución de (53) para alguna $\lambda > 0$.

$\Rightarrow$) Supóngase que $\boldsymbol{\beta}$ minimiza (53) para $\lambda > 0$.

Sea $m = \|\boldsymbol{\beta}\|_0$, $SCE = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$ y $\mathbf{P} = (n, SCE)$. Se mostrará por reducción al absurdo que $\mathbf{P}$ es un punto en la frontera de Pareto.

Supóngase que existe un punto $\mathbf{Q}$ en la frontera tal que $\mathbf{Q} \preceq \mathbf{P}$ ($\mathbf{Q}$ domina $\mathbf{P}$) con $\mathbf{Q} = (n_Q, SCE_Q)$. Por definición de dominancia:

$$SCE_Q + \lambda m_q < SCE + \lambda m$$

$\forall \lambda > 0$.

Entonces $\mathbf{P}$ no minimiza (53)!.

Por lo tanto no existe $\mathbf{Q}$ que domine $\mathbf{P}$, luego $\mathbf{P}$ un punto en la frontera de Pareto.

Para demostrar la proposición inversa, se dará por supuesto que la frontera de Pareto es convexa, en general este supuesto no se cumple, pero nos limitaremos a las que así lo hagan.

Proposición 13.2. *Para cada vector* $\boldsymbol{v}$ *en la frontera de Pareto de* R_G*, que se supondrá convexa, existe al menos una* $\lambda > 0$ *tal que* $\widehat{\boldsymbol{\beta}}$ *con* $\boldsymbol{v} = Soporte(\boldsymbol{\beta})$ *minimiza (53).*

Demostración:

$\Leftarrow$) Sea $R_G = \{\mathbf{P_i} | \mathbf{P_i} = (i, SCE_i)$ para $i = 1, \ldots, m\}$ con $m \leq p$ la frontera de Pareto convexa de (55).

Para fijar ideas, se vé que ocurre si $\boldsymbol{\beta_i}$ y $\boldsymbol{\beta_j}$ tuvieran el mismo valor de acuerdo a (53) para la misma $\lambda > 0$ con $\|\boldsymbol{\beta_i}\|_0 = i$ y $\|\boldsymbol{\beta_j}\|_0 = j$.
Sin pérdida de generalidad $i < j$.

$$\begin{aligned} SCE_j + \lambda j &= SCE_i + \lambda i \\ \lambda j - \lambda i &= SCE_i - SCE_j \\ (j-i)\lambda &= SCE_i - SCE_j \\ \lambda &= \frac{SCE_i - SCE_j}{(j-i)}. \end{aligned}$$

La última ecuación exhibe punto de referencia a partir del cual es más preferible un modelo a otro, de acuerdo a la λ seleccionada. Entonces, sí se quisiera que el modelo con mas variables en el soporte sea la solución, se debería fijar $\lambda < \frac{SCE_i - SCE_j}{(j-i)}$ (pues penaliza el número de variables favoreciendo la suma de cuadrados), y en caso contrario $\lambda > \frac{SCE_i - SCE_j}{(j-i)}$. Con esto en mente se desarrollará la demostración.

$(i = m)$. Supóngase que $\boldsymbol{\beta_i}$ es la estimación del modelo completo.
Como $\|\boldsymbol{\beta_i}\|_0$ toma el valor máximo en la frontera de Pareto, entonces:

$$\lambda \leq \frac{SCE_j - SCE_n}{(n-j)} \quad \forall j < n.$$

Por lo que cualquier λ tal que $0 < \lambda \leq \text{mín}\{\frac{SCE_j - SCE_n}{(n-j)} | j < n\}$ seleccionaría el modelo completo (el modelo con todas las variables en su soporte).

$(i = 0)$. Supóngase que β_i es la estimación del modelo con $\|\boldsymbol{\beta_i}\|_0 = 1$.
Como $\|\boldsymbol{\beta_i}\|_0$ toma el valor mínimo en la frontera de Pareto, entonces:

$$\lambda \geq \frac{SCE_1 - SCE_j}{(j-1)} \quad \forall 1 < j.$$

Por lo que cualquier λ tal que $\lambda \geq \text{máx}\{\frac{SCE_1 - SCE_j}{(j-1)} | 1 < j\}$ seleccionaría el modelo con $\|\boldsymbol{\beta_i}\|_0 = 1$ (el modelo con menos las variables en su soporte).

Hasta esta parte de la demostración no se ha usado el supuesto de convexidad, lo que indica que los modelos completo y de menor cardinalidad en el soporte siempre son seleccionados por alguna λ.

$(0 < i < m)$. Supóngase que $0 < i < m$, entonces sí $\lambda > 0$ minimizara (53) debería cumplirse lo siguiente:

$$SCE_i + \lambda i \leq \min_{j \neq i}\{SCE_j + \lambda j\} \Leftrightarrow \lambda i \leq \min_{j \neq i}\{SCE_j - SCE_i + \lambda j\}$$

$$\Leftrightarrow (i-j)\lambda \leq SCE_j - SCE_i \quad \forall i \neq j$$

$$\Leftrightarrow \begin{cases} \lambda \leq \frac{SCE_j - SCE_i}{(i-j)} & \text{Si } j < i \\ \frac{SCE_j - SCE_i}{(j-i)} \leq \lambda & \text{Si } i < j. \end{cases}$$

Por lo tanto:

$$\text{máx}\{\frac{SCE_k - SCE_i}{(i-k)}\} \leq \lambda \leq \text{mín}\{\frac{SCE_i - SCE_j}{(j-i)}\} \quad \forall j < i < k. \tag{56}$$

¿Qué garantiza que exista alguna λ que cumpla (56)? Procediendo:

Por comodidad se escribirá $f(i)$ para referirse a SCE_i.

Por definición de convexidad:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y) \quad \text{para } 0 \leq t \leq .$$

Sean $j < i < k$, entonces se puede tomar $t = \frac{(k-i)}{(i-j)+(k-i)}$, por lo tanto $1-t = \frac{(i-j)}{(i-j)+(k-i)}$, $x = j$ y $y = k$. Luego:

$$f(i) \leq \frac{(k-i)}{(i-j)+(k-i)}f(j) + \frac{(i-j)}{(i-j)+(k-i)}f(k)$$

$$((i-j)+(k-i))f(i) \leq (k-i)f(j) + (i-j)f(k)$$

$$(i-j)[f(i) - f(k)] \leq (k-i)[f(j) - f(i)]$$

$$\frac{[f(i) - f(k)]}{(k-i)} \leq \frac{[f(j) - f(i)]}{(i-j)} \quad \forall j < i < k.$$

De esta última desigualdad se sigue que el intervalo en el que está contenida la λ necesaria no es vacío.

Resumiendo, la frontera de Pareto de (55) contiene el punto mínimo de (53), cualquiera que sea la λ y en caso de que la frontera sea convexa, ambos conjuntos son idénticos. Así, la $\boldsymbol{\beta}$ que genera cada punto en la frontera se puede entender como la estimación $\boldsymbol{\beta}$ que minimiza la suma de cuadrados de los residuales que contiene a lo mas m variables, con $0 \leq m \leq p$.

$$\widehat{\boldsymbol{\beta}}^{rnsgaii} = \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$$
$$\text{Sujeto a } \|\boldsymbol{\beta}\|_0 \leq m.$$

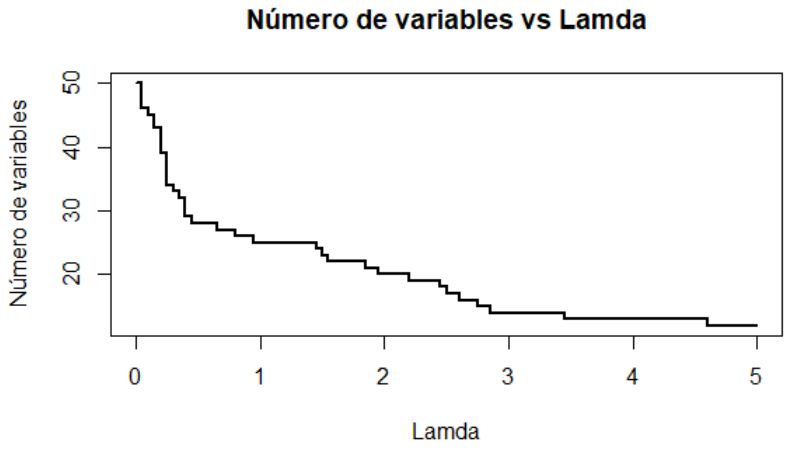

Figura 16: Gráfica de la relación entre la elección del factor $\lambda > 0$ y el número de variables que se encuentran en el modelo que minimiza (53).

13.3. Operadores adaptados

Los operadores que se usarán para el NSGA-II con una codificación binaria son los mas clásicos:

Operador de Selección. Se realizará la selección para cruzamiento queda sin modificación.

Operador de Cruzamiento. El operador de cruzamiento es el descrito en (9.2).

Operador de Mutación. Y las mutaciones como en (9.1).

13.4. NSGA-II adaptado (vista de alto nivel)

Las adaptaciones que se realizaron sobre NSGA-II fueron las siguientes:

Adaptación de los Operadores. Se modificaron los operadores de Cruzamiento y Mutación para hacerlos compatibles con la codificación binaria de los modelos lineales

Creación de una lista tabú. Se crea una lista de modelos tabú con el fin de garantizar que todos los pobladores sean evaluados e introducidos en la población a lo mas una vez. Se justifica en el hecho de que se considera que los cálculos para la obtención de los parámetros de un modelo es mas costoso que realizar la búsqueda, además que le brinda variabilidad a la población.

Hibridación en la inicialización. Se inicializa la población a partir de codificar los modelos producidos por regresiones Lasso, para cada λ (evitando repeticiones).

Con estas modificaciones, el pseudo-código es el siguiente:

1. Generar la población inicial M a partir de codificar las regresiones Lasso.

2. Agregar todos los pobladores a la lista tabú.

3. Evaluar cada elemento en M.

4. Ranquear M (usando el "Ordenamiento rápido no dominado" y la "CrowdingDistance").

5. Seleccionar los r mejores pobladores (con r el tamaño de la población definido por el usuario, auxiliándose del operador Crowded-Comparison $[\prec_n]$).

6. Evaluar la condición de paro, si no se cumple:

6.1. Elegir r individuos con el Operador de Selección por Crowded-Comparison.

6.2. Aplicar el Operador de Cruzamiento para crear una segunda población (Q).

6.3. Aplicar el Operador de Mutación (a los elementos de Q).

6.4. Eliminar los nuevos pobladores (elementos de Q) que se encuentren en la lista tabú.

6.5. Agregar los nuevos pobladores restantes a la lista tabú.

6.6. Combinar la nueva población con la original ($M = M \cup Q$).

6.7. Continuar a partir del punto 3.

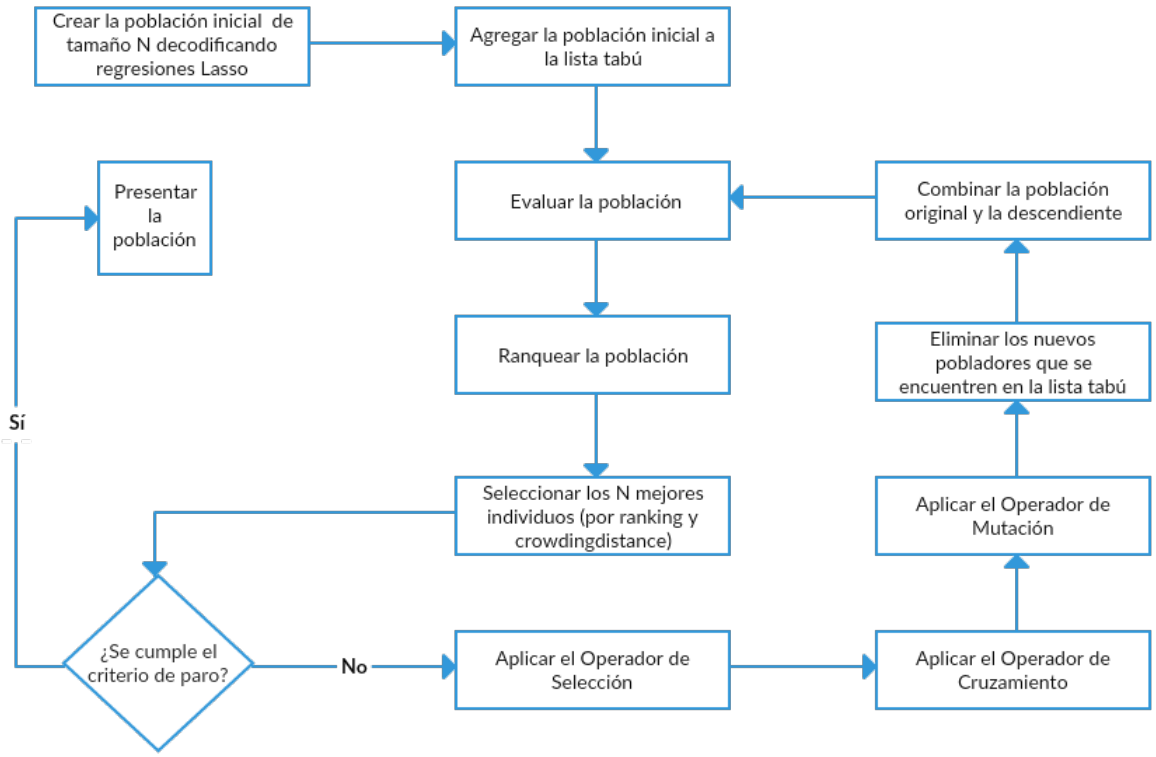

Figura 17: Diagrama de flujo del algoritmo NSGA-II.

13.5. Software

En el apéndice se pone al alcance del lector los diferentes operadores y funciones del algoritmo NSGA-II adaptado (en R versión 3.5.3), encontrándose en orden alfabético. También se halla empaquetando en el archivo $RNSGA2\_1.0.0.tar$ en la liga:

https://drive.google.com/drive/folders/1rVXUq5j_6pADNXnffV1r_WqPrS8nebLU?usp=sharing

Donde podrá descargar un paquete con extensión ".tar" compuesto de 19 funciones, un código de simulación y una tabla (llamada .ªnalisis"[sin acentos]) con estadísticas de éstas, utilizando un software de distribución libre y especializado en computo estadístico llamado **R** (*www.r-project.org*), en la versión 4.0.2

Ahora se mostrará una descripción del contenido del paquete:

Funciones

cruzar: Genera los pobladores hijos a partir de dos pobladores.

extinsion: Realiza las acciones del operador de reemplazo. Selecciona los pobladores que sobreviven (continúan en la población) entre la población original y su descendencia.

extraer_adn: Codifica un modelo de la clase "lm" como un vector binario.

$formular$: Decodifica un vector binario a un objeto de la clase "formula", que es el paso es un insumo para calcular un objeto "lm".

generar_datos: Genera una matriz con observaciones de un plano de regresión simulado.

iniciar_poblacion: Imitando a $Lass-0$, *iniciar_poblacion* calcula las estimaciones de la regresión Lasso y las codifica para tomarlas como punto de arranque. Si la población es de un tamaño mayor a los elementos que se pueden obtener por Lasso; se mutan los logrados hasta alcanzar el tamaño poblacional.

instalar_paquetes: Instala los paquetes de los que depende $RNSGA2\_1.0.0.tar$, sólo si no se encuentran. En otro caso no realiza alguna acción al respecto.

modelar: Decodifica un vector binario a un objeto de la clase "lm".

mutar: Realiza las funciones del operador de mutación.

panel: Crea un histograma y una gráfica de dispersión la función objetivo sobre la población en formato de panel.

rangos: Es una función que cambia un tipo "list" a tipo vector la lista que contiene el ranking de cada poblador.

reemplazo_secuencial: Realiza una búsqueda local de modelos óptimos con el mismo número de variables partiendo de un vector binario. Más adelante se le dedica espacio para explicar el método.

Funciones Objetivo

Las restantes funciones y el código de simulación se merecen una explicación mas detallada por el impacto que pueden tener en los resultados del algoritmo. Pero si se desean más detalles de las ya presentadas, se puede dirigir a la ayuda contenida dentro del mismo paquete. El código está comentado en español, omitiendo tildes para evitar problemas de compatibilidad con codificaciones no compatibles con el español.

Funciones objetivo. De acuerdo a (55) la función objetivo "canónica" se puede expresar de la siguiente forma: Donde *adn* es la codificación binaria y *datos* la matriz de datos.

Pero gracias a la flexibilidad de los algoritmos genéticos es posible reducir el espacio de búsqueda (equivalente a imponer restricciones a la función objetivo) sin implicar un cambio en la representación. Por ejemplo:

Es común querer especificar si el modelo a estimar cuenta o no con intercepto, así, se puede penalizar los modelos sin intercepto (o con el) con un valor que sea peor que todos el rango de la función objetivo. **R** cuenta con el valor Inf para expresar esta situación. Cuando λ que tiene un valor que se pueda considerar despreciable, la función anterior posee un impacto relevante, pues el algoritmo arroja los modelos de regresión Lasso de menor soporte. Importante pues la estimación Lasso no siempre es única, aunque ese es un tema que va mas lejos que los objetivos de esta tesis. Véase Tibshirani, Ryan J. (2013).

Una observación interesante es que en ninguna de las funciones objetivo cuenta los ceros de la estimación, ni siquiera en Lasso. Esto está justificado, pues si se estimara una variable con el valor exacto de cero, entonces existiría una codificación con un cero (al menos) más. Así que la evolución se encarga de que los soportes coincidan.

Lo siguiente es explicar la función principal, $RNSGAII$. Esta función se encarga de coordinar los demás operadores y funciones para poder emular la evolución. Y cuenta con trece parámetros que se explican a continuación.

Parámetros de la función RNSGAII

Datos: Es la matriz que contiene las observaciones del plano de regresión, tanto la dependiente (**Y**), como las explicativas.

cl: Asigna un número de núcleos de procesamiento para la ejecución paralelizada del algoritmo.

$funcion.fit$: Texto con el nombre de la función objetivo.

hibridar: Es un valor entero que indica la ciclicidad de la hibridación del algoritmo genético con el algoritmo de reemplazo secuencial (se aplica a toda la población). Hibridar ayuda a disminuir el tiempo de convergencia del algoritmo. Pero por default, se asigna Inf, lo que significa que no hay hibridación.

indxtorneo: Número de individuos de la población involucrados en la selección por torneo. Por default se asigna un 10 %.

iteraciones: Número máximo de generaciones.

parametros Vector de textos que representan objetos que se replicarán en los núcleos de procesamiento para poder ejecutar la función de valuación.

poblacion_i: Parámetro que contiene una matriz binaria. Este parámetro permite inicializar manualmente la población.

pmg: Probabilidad de mutación de un gen, se considera constante para todos los genes. Por default se le asigna $1/dim(datos)[2]$, para que en esperanza se modifique un sólo gen.

psm: Probabilidad de selección de mutación. Por default se mantiene en un 5%.

n_mdls: Tamaño de la población.

$Tparo1$: Número de segundos máximo que puede durar el algoritmo.

$Tparo2$ Número de generaciones en las que si no hay un cambio, se termina la ejecución.

En resumen, lo primero que nos planteamos en la función objetivo considerando las restricciones necesarias, se eligen los parámetros del algoritmo genético (probabilidad de selección, probabilidad de mutación, tiempo de paro, tamaño de población, número de individuos por torneo, etc.). Se inicializa la población, puede ser aleatoria o mediante algún otro método, esto se justifica ya que la convergencia del $RNSGAII$ esta garantizada por su elitismo (la primera frontera de Pareto es un estado absorbente).

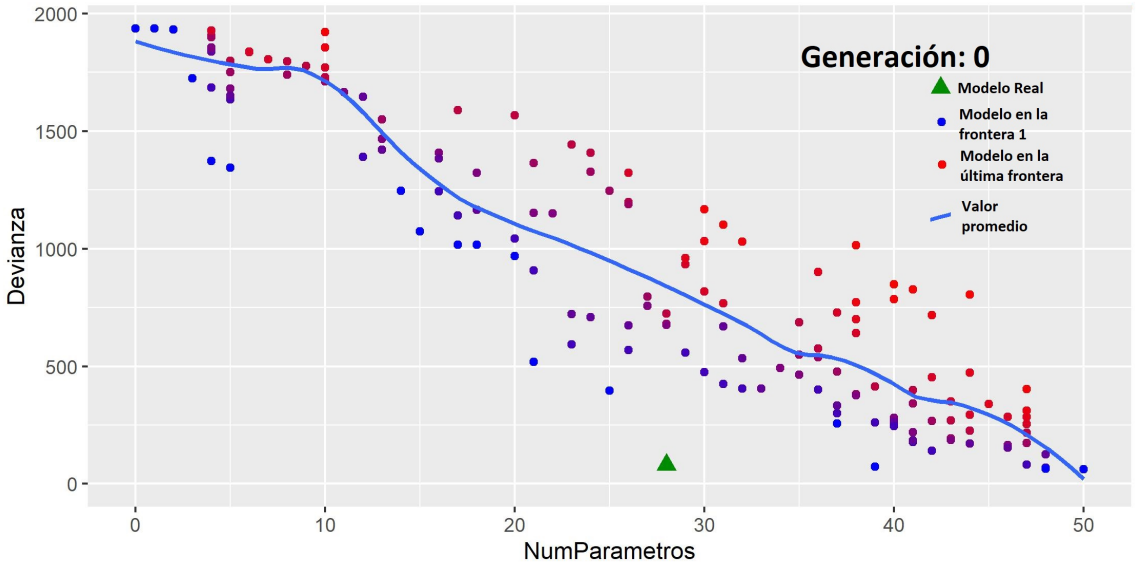

Figura 18: Gráfica de una población de modelos de regresión lineal inicializada aleatoriamente.

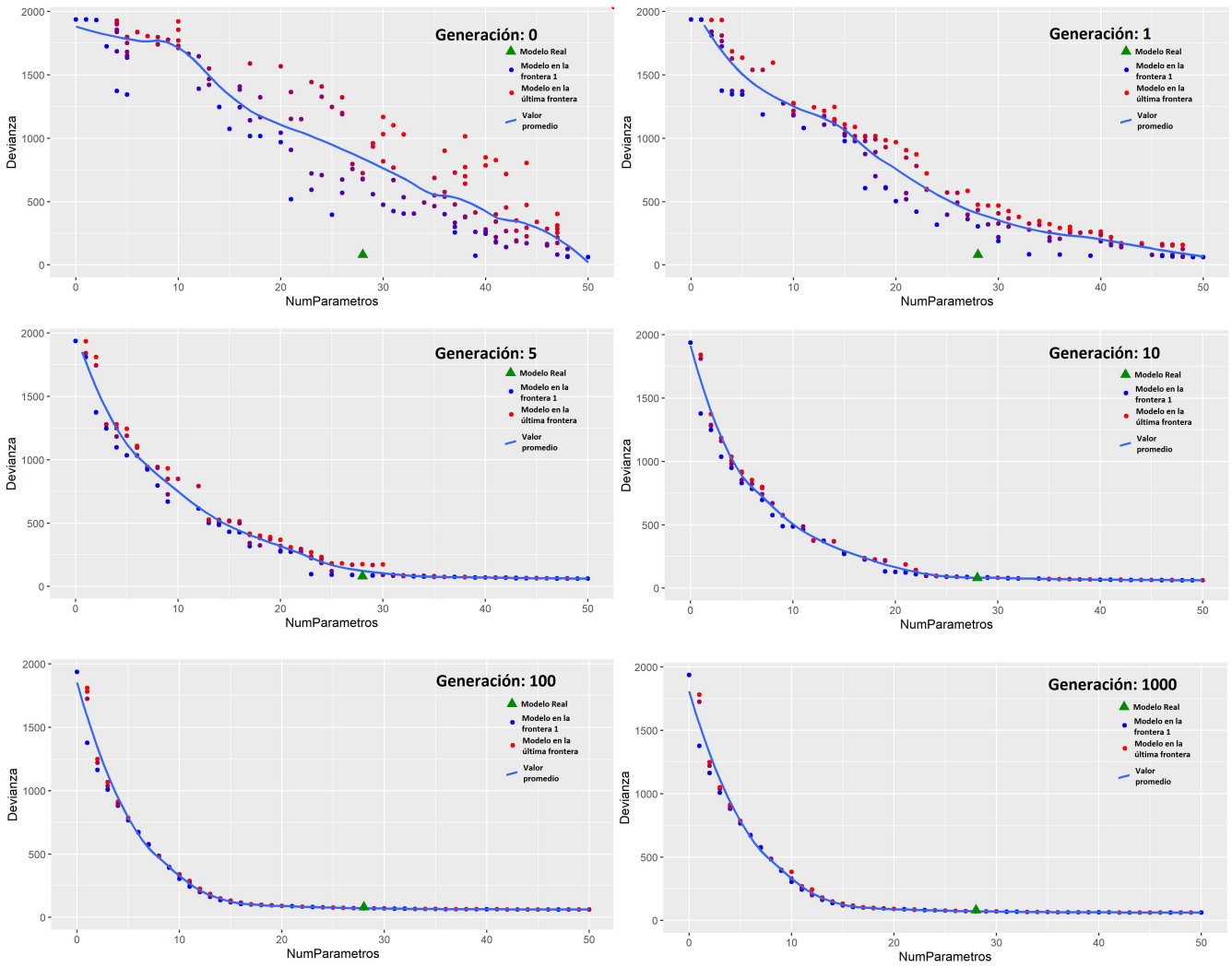

Figura 19: Convergencia del *RNSGAII*.

Aún falta explicar el código para el estudio de simulación, pero esto se hará hasta el último capítulo.

Selección de modelos

Como se mencionó anteriormente, en investigación cuantitativa se puede tener diferentes objetivos respecto del proceso de análisis de datos, puede ser un objetivo predictivo, descriptivo o de contraste de hipótesis, pero en todos los casos se debe contar con uno o varios modelos que cumplan características específicas para alcanzar la meta de investigación.

Ahora, encontrar tales modelos puede no ser una tarea sencilla pues (en general) conforme el número de variables predictivas crece, el número de posibles modelos crece exponencialmente. Particularmente en análisis de regresión lineal, cuando los datos están formados por p variables explicativas, se tienen $2^p - 1$ posibles modelos.

Con la finalidad de atacar esta dificultad se han creado multitud de técnicas y criterios para apoyar a seleccionar el (los) modelo(s) mas adecuado(s). En esta parte de la presente tesis se tratarán los criterios de selección de modelo mas conocidos y su relación con el algoritmo que se ha propuesto en el capítulo anterior.

14. Criterios de selección

Un criterio (o procedimiento) de selección es una función $C : \mathbb{M} \times D \to \mathbb{R}$ con $\mathbb{M}$ el conjunto de todos los modelos de regresión lineal múltiple dadas n observaciones de la variable dependiente $\mathbf{Y}$ y sus covariables $\mathbf{X}_1, \ldots, \mathbf{X}_p$ (se agruparán estas variables en una matriz que se nombrará con la letra D). Se dice que C selecciona un modelo $\widehat{M}$ si:

$$\widehat{M} = \arg\min_{M \in \mathbb{M}} C(M, D).$$

Los criterios de selección son métodos que asignan una valoración (en $\mathbb{R}$) a cada modelo de $\mathbb{M}$ de acuerdo a sus características e información disponible. De esta forma, se trata de aplicar un principio global a todos los modelos para inducir un orden de preferencia (auxiliándose del orden de $\mathbb{R}$). Así, un criterio crea orden pero no indica o reduce el espacio de búsqueda del modelo que selecciona, mientras que los procedimientos de selección sí establecen el algoritmo de búsqueda.

Lo que se verá a continuación son algunos de los criterios de selección aplicados a la elección de modelos de regresión lineal, supondrá que $\mathbf{X}'\mathbf{X}$ es una matriz diagonal y con varianza ($\widehat{\sigma}^2$) de los residuales desconocida.

14.1. AIC

El AIC, o criterio de Akaike, fue propuesto por Hirotsugu Akaike en 1970, y se calcula como a continuación:

$$\begin{aligned} AIC &= 2k - log(f) && \text{(Expresión general)} \\ &= \frac{n}{2} log(\frac{SCE}{n}) + k && \text{(Expresión para modelos lineales).} \end{aligned} \tag{57}$$

Donde:

- k es el número de variables explicativas ($k \leq p$).
- f es la verosimilitud del modelo dados los datos.
- n el número de observaciones.
- SCE es la suma de cuadrados de los residuales.

14.2. BIC

El BIC es un criterio de selección de modelos propuesto por Gideon E. Schwarz desde un punto de vista bayesiano. Está definido de acuerdo a la próxima ecuación:

$$\begin{aligned} BIC &= klog(n) - 2log(f) && \text{(Expresión general)} \\ &= \frac{n}{2} log(\frac{SCE}{n}) + \frac{k}{2} log(n) && \text{(Expresión para modelos lineales).} \end{aligned} \tag{58}$$

14.3. Coeficiente R^2 y R^2 ajustado

El coeficiente de determinación múltiple (o R^2) es un indicador calculado a partir de la descomposición de la suma de cuadrados (12) que mide la proporción de la varianza que es explicada por el modelo de regresión:

$$R^2 = \frac{SCR}{SCT} = 1 - \frac{SCE}{SCT}. \tag{59}$$

Donde SCR y SCE representan la suma de cuadrados de la regresión y de cuadrados totales, expresadas en las ecuaciones (13) y (14) respectivamente.

Es inmediato notar $0 \leq R^2 \leq 1$. En este caso el criterio es de maximización (el más cercano a 1 es el mejor), aunque hay que tener en cuenta que el modelo completo siempre es el que minimiza la suma de cuadrados de los errores (SCE), por lo que no penaliza la complejidad del modelo (medida como por el número de variables).

El R^2 ajustado es definido como:

$$R^2_{ajustado} = 1 - (\frac{n-1}{n-p-1}) \frac{SCE}{SCT}.$$

El R^2 ajustado es una transformación del R^2 que busca penalizar el número de regresores para obtener modelos más parsimoniosos.

14.4. C_p de Mallows

El C_p de Mallows fue propuesto por Colin Lingwood Mallows como:

$$C_p = \frac{SCE}{\widehat{\sigma}^2_{Completo}} - (n - 2k).$$

Donde $\widehat{\sigma}^2_{Completo}$ es el estimador de la varianza de los residuales del modelo con todas las variables.

El AIC, BIC, Cp de Mallows seleccionan el modelo que minimice la expresión mientras que el R^2 y $R^2_{ajustado}$ eligen maximizandola.

En Foster, D. P. and George, E. I. (1994) se propone una forma canónica de los criterios de selección de variables, en el caso donde la varianza de los residuales es desconocida (σ^2) como:

$$\gamma_\pi = \arg\min_{\gamma\in\Gamma}[SCE_\gamma + k\sigma^2\widehat{\pi}].$$

Donde $\widehat{\pi}$ es una penalización que depende de los datos.

Y se muestra que tanto el AIC, BIC y C_p se pueden reescribir en la forma canónica, se verá el C_p como ejemplo:

$$C_p = \widehat{\sigma}^{-2}[SCE - k\widehat{\sigma}^{-2}2] - n.$$

Donde $\widehat{\pi} = 2\frac{\widehat{\sigma^2}}{\sigma^2}$, que claramente $\widehat{\pi} \to 2$ cuando $n \to \infty$.

De manera similar se puede encontrar el parámetro que sanciona el número de parámetros del AIC y BIC:

AIC**:** $\widehat{\pi}_{AIC} = 2 + O[(\frac{SCE}{n\sigma^2}) - 1]^2$.

BIC**:** $\widehat{\pi}_{BIC} = log(n) + O[(\frac{SCE}{n\sigma^2}) - 1]^2$.

Así, se puede ver que $\widehat{\pi}_{AIC} \to 2$ y $\widehat{\pi}_{BIC} \to log(n)$.

Sí se observa la función de perdida de la regresión dispersa no convexa (53) y la forma canónica de los criterios de selección de variables se observa que tienen la misma forma. Y como se mostraba en (13.1), todas las soluciones a (53) se encuentran en la frontera de Pareto de la función (55). Entonces la conclusión es que la frontera de Pareto debe contener los mejores modelos (de acuerdo a los criterios de la forma canónica).

La conclusión anterior es muy natural, pues desde el punto de optimización multiobjetivo, la forma canónica es parte de las estrategias basadas en agregación que se mencionaban en la sección de los operadores de selección. Ahora, es de interés saber si la frontera de Pareto

que se cálculo con NSGA-II contiene otra clase de criterios de selección. Para poder aclarar esto se escribe la siguiente definición.

Definición 14.1. *Buen criterio en el sentido de Pareto*

Se dice que $C : \mathbb{R}^+ \times \mathbb{N} \to \mathbb{R}$ *es un buen criterio o procedimiento en el sentido de Pareto (o sólo buen criterio) si cumple que:*

- *Para cualquier* k_0 *fija, con* $k_0 \in \{0, \ldots, p\}$, $C(SCE, k_0)$ *es una función creciente.*

Donde p *es el número de covariables en los datos,* k *es la norma cero de los parámetros del modelo a evaluar y* SCE *la suma de cuadrados de los errores.*

Ahora se mostrará algunos ejemplos de "buenos criterios", usando derivadas parciales para mostrar que cumplen la definición:

- Criterios de la forma canónica.

$$\frac{\partial \gamma_\pi}{\partial SCE} = 1 > 0.$$

- El AIC con σ^2 desconocida y n fija.

$$\frac{\partial AIC}{\partial SCE} = \frac{n^3}{2SCE} > 0.$$

- El BIC con σ^2 desconocida y n fija.

$$\frac{\partial BIC}{\partial SCE} = \frac{n^3}{2SCE} > 0.$$

- El $R^2_{ajustado}$.

  En este caso se calcula para $1 - R^2_{ajustado}$, pues se definió "criterio de selección de variables" desde el punto de vista de maximización.

$$\begin{aligned} 1 - R^2_{ajustado} &= (\frac{n-1}{n-k-1})\frac{SCE}{SCT} \\ &= (\frac{n-1}{n-k-1})\frac{SCE}{SCR+SCE}. \end{aligned}$$

  Luego:

$$\frac{\partial (1 - R^2_{ajustado})}{\partial SCE} = \frac{n^3}{2SCE} > 0.$$

- El criterio Amemiya (Amemiya, T. 1976):

  El criterio se calcula como:

$$\frac{n+k}{n-k}(1 - R^2).$$

Donde R^2 es el coeficiente R cuadrado. Ó equivalentemente:

$$(\frac{n+k}{n-k})\frac{SCE}{SCR+SCE}.$$

Entonces

$$\frac{\partial Amemiya}{\partial SCE} = (\frac{n+k}{n-k})\frac{SCR}{(SCR+SCE)^2} > 0.$$

- La prueba F contra el modelo de todas las covariables.
  Sólo para tener a la mano el estadístico de la prueba F se reescribe como:

$$F = (\frac{n-k-1}{k})\frac{SCE_0 - SCE_A}{SCE_A}.$$

  Donde SCE_0 es la suma de cuadrados bajo la hipótesis nula (bajo el modelo reducido) y SCE_A bajo la alternativa. Entonces:

$$\frac{\partial F}{\partial SCE_0} = (\frac{n-k-1}{k})\frac{1}{SCE_A} > 0.$$

Proposición 14.1. *La frontera de Pareto de la función objetivo (55), bajo los supuestos del modelo de regresión lineal, está compuesta (en la mayoría de los casos) por todos los modelos que son seleccionados por algún buen criterio en el sentido de Pareto y sólo por estos.*

Demostración: Se demostrará que la frontera de Pareto de (55) contiene todos los modelos que pueden ser seleccionados por algún buen criterio.

⇒) Sea M un modelo en la frontera de Pareto. Hay que demostrar que M es elegido por algún buen criterio.

Sea $\widehat{\beta}_M$ la estimación asociada al modelo M. Como M es un modelo de la frontera, se cumple que:

$$\widehat{\boldsymbol{\beta}}_M = \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$$
$$\text{Sujeto a } \|\boldsymbol{\beta}\|_0 = k_0.$$

Sea C un buen criterio en el sentido de Pareto. Entonces:

$$M = \arg\min_{M\in\mathbb{M}} C(\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_M\|_2^2, \|\widehat{\boldsymbol{\beta}}_M\|_0)$$

Ya que $C(SCE, k_0)$ es creciente para cualquier k_0 fija.

Por lo tanto, todos los puntos de la frontera de Pareto son elección de al menos un buen criterio.

⇐) Sea $\widehat{M}$ la elección de un buen criterio C. Se demostrará por reducción al absurdo que $\widehat{M}$ es un punto en la frontera de Pareto.

Sea $\widetilde{M}$ un modelo que domine a $\widehat{M}$ ($\widetilde{M} \prec \widehat{M}$) y sean $\widehat{\boldsymbol{\beta}}_{\widetilde{M}}$ y $\widehat{\boldsymbol{\beta}}_{\widehat{M}}$ las estimaciones asociadas a los modelos $\widehat{M}$ y $\widetilde{M}$ respectivamente.

Como $\widetilde{M} \prec \widehat{M}$, entonces $\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widetilde{M}}\|_2^2 \leq \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widehat{M}}\|_2^2$ y $\|\widetilde{M}\|_0 \leq \|\widehat{M}\|_0$.

Viendo que no puede ocurrir la igualdad en el número de regresores:

Ya que en ese caso $\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widetilde{M}}\|_2^2 < \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widehat{M}}\|_2^2$ con el mismo número de variables. Luego $\widetilde{M}$ sería la elección de C, pues es creciente!.

Por lo tanto $\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widetilde{M}}\|_2^2 \leq \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widehat{M}}\|_2^2$ y $\|\widetilde{M}\|_0 < \|\widehat{M}\|_0$.

Como $\|\widetilde{M}\|_0 < \|\widehat{M}|_0$ entonces, en la mayoría de los casos se puede buscar un modelo $\widetilde{M}^*$ que sea anidado con $\widetilde{M}$ y que $\|\widetilde{M}^*\|_0 = \|\widehat{M}\|_0$, es decir, se forma a $\widetilde{M}^*$ de agregar variables a $\widetilde{M}$ hasta alcanzar el número de $\widehat{M}$. Los casos donde no se puede encontrar es cuando al agregar variables la estimación por mínimos cuadrados los calcula como cero.

Entonces:

$$\begin{aligned} \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widetilde{M}^*}\|_2^2 &< \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widetilde{M}}\|_2^2 \qquad (*) \\ &\leq \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widehat{M}}\|_2^2 \qquad \text{(por ser C creciente).} \end{aligned}$$

$*$ En la mayoría de las ocasiones y por la unicidad del estimador de mínimos cuadrados (el soporte de $\widetilde{M}^*$ no se puede tener dos $\boldsymbol{\beta}$'s que minimicen la suma de cuadrados de los errores).

Por lo tanto $\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widetilde{M}^*}\|_2^2 < \|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\widehat{M}}\|_2^2$.

Por lo que se concluye que la mayoría de las ocasiones $\widehat{M}$ estará en la frontera de Pareto.

De todos los casos se concluye la igualdad de los conjuntos (en la mayoría de los casos).

Así, la proposición anterior aclara que clase de modelos se puede encontrar en la frontera de Pareto de nuestra función objetivo. Otra pregunta natural sería, ¿es posible agregar mas entradas a (55)?, la respuesta es que sí, pero sólo tiene sentido agregar funciones que no sean "buenos criterios", ya que de lo contrario no habría cambio en la frontera.

Regresando al ejemplo de la prueba F, los modelos de la frontera de Pareto bajo esta estrategia de selección, se pueden interpretar como los modelos que minimizan la probabilidad de rechazar la hipótesis nula.

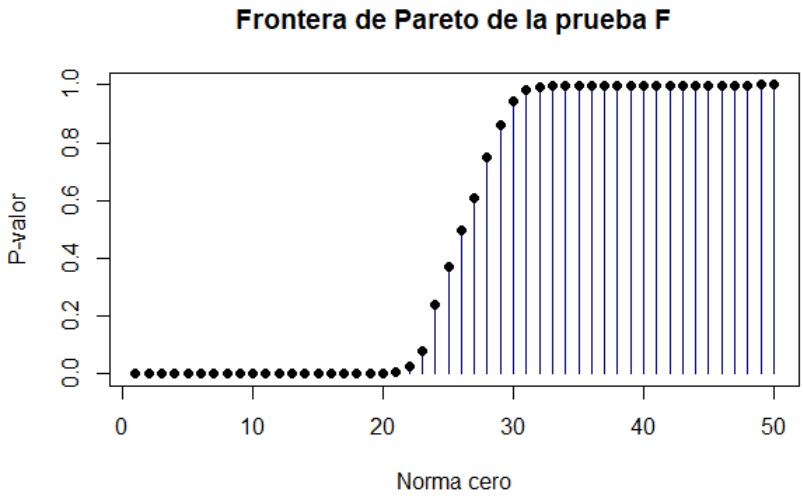

Figura 20: Frontera de Pareto de la prueba F. Ejemplo simulado en el software R versión 3.5.3.

15. Algoritmos de búsqueda de modelos

En esta sección se tratara algunos de los algoritmos de construcción de modelos de regresión lineal y su relación con la frontera de Pareto.

15.1. Backward

Backward, en términos generales, es un algoritmo que simplifica modelos con un buen ajuste para terminar en uno que simplifique el primero y conserve el ajuste en algún rango. El método comienza con un modelo que ofrezca una suma de cuadrados relativamente pequeña y partiendo de esto, ir simplificando el modelo (quitando regresores), perdiendo el menor desempeño posible en la muestra. Así, el algoritmo termina en a lo mas en p iteraciones (p el número total de regresores). En la siguiente lista se muestra el algoritmo.

1. Definir el criterio de selección.

2. Definir la condición de paro.

3. Calcular M_k, un modelo con $k > 0$ regresores, por default $k = p$.

4. Para $j = k$ hasta que se cumpla la condición de paro ó se llegue al modelo vacío (con 0 regresores), hacer lo siguiente:

4.1. Calcular los j modelos, que contienen todas las variables de M_j excepto una.

4.2. Escoger el mejor de modelo de acuerdo al criterio establecido, llamarlo M_{k-1}.

5. Regresar el modelo $M_{k_{min}}$, donde $k_{min} = \text{mín}\{j | j \leq k\}$.

15.2. Forward

De manera dual a Backward, Forward comienza con un modelo simple, y se agregan regresores hasta alcanzar la condición de paro. La idea es lograr un buen ajuste agregando sólo las mejores variables. De igual manera, el algoritmo termina en a lo mas p iteraciones. A continuación los pasos de Forward.

1. Definir el criterio de selección.

2. Definir la condición de paro.

3. Calcular M_k, modelo con $k < p$ regresores, por default $k = 0$.

4. Para $j = k$ hasta que se cumpla la condición de paro ó se llegue a p, hacer lo siguiente:

4.1. Calcular los $k - j$ modelos, que contienen todas las variables de M_j más una.

4.2. Escoger el mejor de modelo de acuerdo al criterio establecido, llamarlo M_{j+1}.

5. Regresar el modelo $M_{k_{max}}$, donde $k_{max} = \text{máx}\{j | j \geq k\}$.

Como se mencionó, cuando $\mathbf{X}'\mathbf{X}$ es diagonal, la solución que presentan en Goran Marjanovic, Magnus O. Ulfarsson, Alfred O. Hero III (2015) se presenta una solución a (53), que se reescribe para hacer una observación.

$$\widehat{\boldsymbol{\beta}}_j^{N_0} = \begin{cases} \mathbf{X}_j\mathbf{Y} & \text{si } \mathbf{X}_j\mathbf{Y} > \sqrt{\lambda} \\ 0 & \text{si } |\mathbf{X}_j\mathbf{Y}| \leq \sqrt{\lambda} \\ \mathbf{X}_j\mathbf{Y} & \text{si } \mathbf{X}_j\mathbf{Y} < -\sqrt{\lambda}. \end{cases}$$

En la expresión anterior se observa que el número de variables estimadas con cero depende de $\sqrt{\lambda}$, que es una función estrictamente creciente. Luego, sí se incrementa el valor de λ; se quitan variables del soporte y no vuelven a entrar para cualquier valor mayor a la λ actual. Entonces, el conjunto de soluciones (que coincide con la frontera de Pareto) es un grupo de modelos que están anidados.

Así, cuando $\mathbf{X}'\mathbf{X}$ es diagonal, se tiene la garantía de que los modelos de la frontera de Pareto están anidados (es decir, si se toma un par de modelos, el de menos variables se encuentra anidado en el de más).

Como se puede notar, tanto Backward como Forward, construyen conjuntos de modelos anidados que por lo normal comienzan de un modelo en la frontera de Pareto (el modelo con todos los regresores y sin regresores están en la frontera por ser el más simple y el de mejor ajuste, respectivamente). Y si además se usa un buen criterio en el sentido de Pareto (como lo es la prueba F), se tiene la conclusión que ambos algoritmos coinciden con la frontera de Pareto.

La conclusión anterior es muy natural al comparar los algoritmos mencionados con $Lass-0$, pues se observa que los pasos de $Lass-0$ coinciden con Backard o Forward si toman un mismo punto de partida.

Ahora se verá el algoritmo propuesto por Efroymson, M. A. en 1960, más conocido como "Stepwise".

15.3. Stepwise

Stepwise es un algoritmo que mezcla los principios de los dos anteriores. De igual manera que Forward, empieza con un modelo simple y agrega variables hasta construir un modelo adecuado, pero cada iteración revisa que todas las variables aporten al modelo y de ser necesario quita alguna. La idea es poder corregir variables que en iteraciones anteriores fueron candidatas viables para el modelo final, pero que en presencia de otras ya no lo sean.

Aunque en general, Stepwise engloba un grupo de algoritmos para la elección de modelos, aquí se mostrará el procedimiento basado en la prueba F.

1. Definir dos p-valores, uno de entrada $0 < \alpha_e < 1$ y otro de salida $0 < \alpha_s < 1$.

2. Calcular M_k, modelo con k regresores ($k \leq p$ donde p es el número total de variables), por default se inicializa $k = 0$ (sin covariables).

3. Evaluar si alguna variable que no se encuentre en M_k es significativa con un nivel α_e. Sí no se encuentra; la solución del algoritmo es M_k. En caso contrario, continuar.

4. Calcular M_{k+1}, el modelo anidado a M_k agregando la variable con el menor estadístico F:

$$F = \frac{SCE_k - SCE_{k+1}}{SCE_{k+1}/[n-(k+1)-1]}.$$

5. Evaluar si alguna variable que se encuentre en M_{k+1} no es significativa con un nivel α_s. Sí es el caso, eliminar la variable con menor estadístico F (se llamará a este modelo M_k^*), luego hacer $M_k = M_k^*$. En caso contrario hacer $k = k+1$.

$$F = \frac{SCE_k - SCE_{k+1}}{SCE_{k+1}/[n-(k+1)-1]}.$$

6. Regresar al punto 3.

15.4. Reemplazo secuencial

El reemplazo secuencial es un método de búsqueda local donde se parte de un modelo (M) de k variables explicativas de un total de p. Para cada variable explicativa (i) y por cada variable exógena a M, se crea un modelo con que sólo difiera a M por la variable número i (que es reemplazada por una variable exógena). Se revisa si alguno de los modelos tiene menor SCE que M y en caso afirmativo se toma como modelo inicial y se repite el proceso hasta que ya no se pueda mejorar la SCE.

Se aclararán estas ideas al ver el pseudo-código.

Sean $\mathbf{X}_1, \ldots, \mathbf{X}_p$ las covariables contenidas en los datos.

Sin pérdida de generalidad, sea $M :=< Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k >$ el modelo inicial.

1. Calcular la suma de cuadrados de los residuales del modelo SCE_M.

2. Inicializar $i = 1$.

3. Calcular la suma de cuadrados de los errores $SCE_{i,j}$ para cada par X_i (variable endógena) y X_j (variable exógena).
Donde:

$$M_{i,j} :=< Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_{i-1} x_{i-1} + \beta_j x_j + \beta_{i+1} x_{i+1} + \cdots + \beta_k x_k > .$$

4. Sí $SCE_M > min\{SCE_{i,j} | j > k\}$ entonces:

$$M = \arg\min\{SCE_{i,j} | SCE(M_{i,j})\}.$$

.

5. Sí hubo un cambio de modelo entonces regresar al paso 3. En caso contrario se ha terminado.

15.5. Búsqueda exhaustiva

Como se mencionó al principio de la sección, el número de modelos posibles crece exponencialmente con cada variable, específicamente para 50 variables se tendrían 1,125,899,906,842,623 posibles modelos. Claramente no es viable hacer una búsqueda por fuerza bruta con una computadora de escritorio en un espacio de miles de billones de modelos o mayor.

La forma mas común de atacar este problema es adaptar el algoritmo de ramificación y poda al contexto de regresión, ejemplos de esto son Beale et al. (1967), Hocking y Leslie (1967), LaMotte y Hocking (1970) entre otros.

La mayoría de las estrategias se guían por dos hechos:

- Para modelos anidados la suma de cuadrados es decreciente respecto del número de variables:

$$SCE(M_1) \leq SCE(M_2) \iff \|\boldsymbol{\beta}_{M_1}\|_0 \geq \|\boldsymbol{\beta}_{M_2}\|_0.$$

- En general, no todas las variables aportan a la variable dependiente ($\mathbf{Y}$) y ordenarlas apoya a evaluar primero a los mejores modelos.

Observando como funciona:

Supóngase que se cuenta con $\mathbf{X}_1, \ldots, \mathbf{X}_p$ variables. Y se desea encontrar el mejor modelo de p variables.

Se puede empezar a clasificar los modelos por aquellos que contienen la variable $\mathbf{X}_1$ y los que no, en este caso, la división crea dos ramas. De igual manera se puede volver a organizar los modelos en los que contienen $\mathbf{X}_2$ de las que no, y así sucesivamente. Cortando cada rama cuando se ha llegado al punto donde sus elementos cuentan con p variables explicativas.

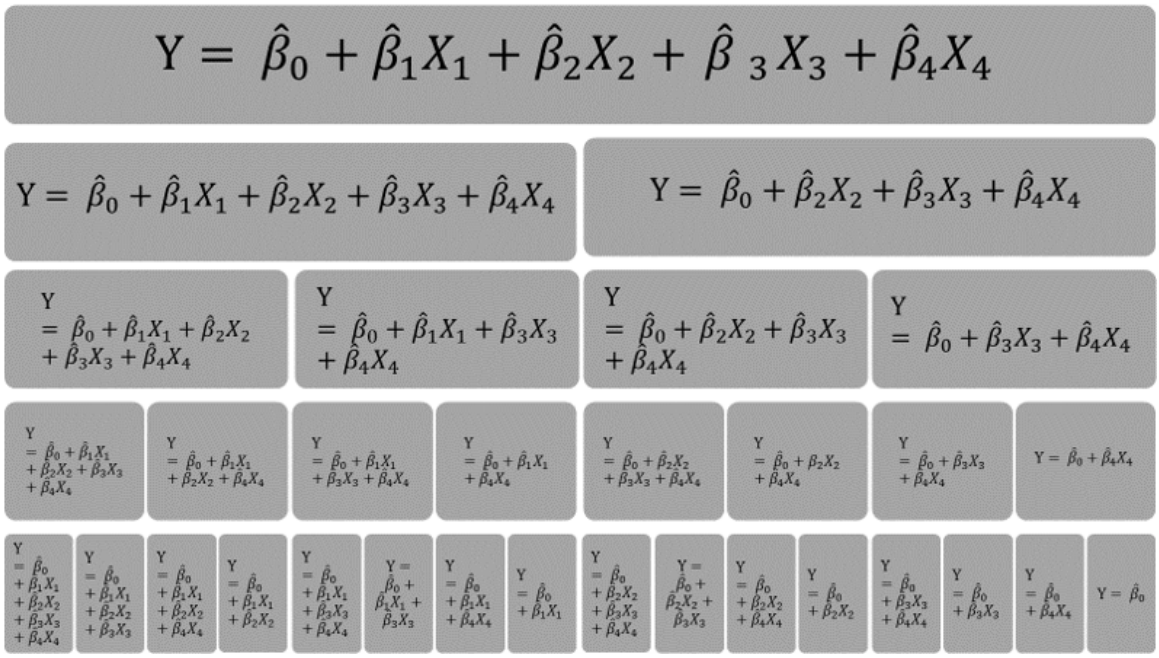

Figura 21: Ramificación de todos los modelos con constante para cuatro variables explicativas.

Ahora supóngase que se han valuado modelos (de izquierda a derecha) llegado al punto de analizar la sub-rama donde se encuentran los modelos sin el regresor $\mathbf{X}_1$, entonces en esta rama la suma de cuadrados de residuales son mayores o iguales a la de $SCE(X_2, \ldots, X_p)$. Luego, sí ya se ha encontrado un modelo M tal que $SCE(\mathbf{X}_2, \ldots, \mathbf{X}_p) > SCE(M)$ hace innecesario evaluar la sub-rama donde no se encuentre la variable $\mathbf{X}_1$.

En el siguiente ejemplo, se supondrá que se cuentan con cuatro variables explicativas y se desea encontrar el modelo que minimice la suma de cuadrados. El proceso seria el siguiente:

1. Se ordenan los modelos en forma de árbol.
2. Se evaluan las SCE de los modelos que se encuentren en las hojas del árbol, de izquierda a derecha.
3. Si en algún momento se encuentra un par de modelos, M_a y M_b, donde $SCE(M_a) < SCE(M_b)$ donde el primero tenga menos variables que el segundo; se puede cortar la rama del segundo.

   Por ejemplo:

   $$M_8 :=< Y = \widehat{\beta}_0 + \widehat{\beta}_1 X_1 >$$
   $$M_9 :=< Y = \widehat{\beta}_0 + \widehat{\beta}_2 X_2 + \widehat{\beta}_3 X_3 + \widehat{\beta}_4 X_4 >$$

   Donde $SCE(M_8) < SCE(M_9)$. Entonces ya no será necesario calcular la SCE para los modelos anidados con M_9.

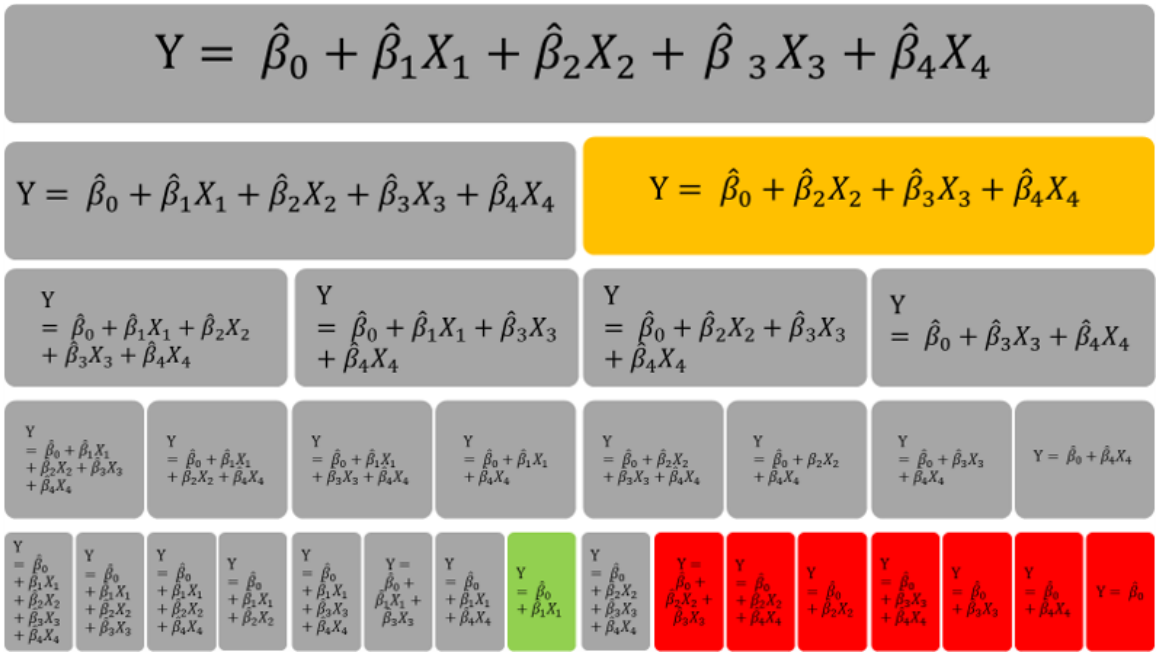

Figura 22: Ejemplo de poda en el árbol de modelos con constante para cuatro variables explicativas.

Simulaciones

Para comprobar los resultados se generaron datos simulados de modelos de regresión lineal múltiple que cumplen las siguientes características:

- El número de covariables en la matriz de datos es 50.
- Se fija el número de observaciones en 100.
- Se simula el soporte de la regresión "real" como un vector de observaciones $Bernoulli(u)$ con $u \sim Unif(0,1)$, para obtener modelos donde cualquier número de variables sea posible con igual probabilidad.
- Se simulan las pendientes de las variables en el soporte como observaciones de una $Normal(0,1)$.
- Las observaciones de las variables explicativas se simulan de una $Normal(\mathbf{0}_{100}, \mathbf{\Sigma})$.

  Donde $\mathbf{0}_{100}$ es el vector de medias de tamaño 100 y $\mathbf{\Sigma}$ es la matriz de varianzas-covarianzas.

  Se toma la $\mathbf{\Sigma} = \mathbf{I}_{p\times p}$ (con p el número de variables) ó se construye como $\mathbf{\Sigma} = \mathbf{A}'_{p\times p}\mathbf{A}_{p\times p}$ donde las entradas de $\mathbf{A}_{p\times p}$ se obtienen de una distribución $Cauchy(0,1)$ (ya que de esta forma se obtienen correlaciones bajas, medias y altas) y se estandariza (se ajustan las varianzas igual a 1).
- Cuando se calcula la variable dependiente, se centra en cero si el modelo no tiene intercepto.

Se cuida que la proporción entre los modelos con intercepto y sin intercepto sean iguales. Se tiene la misma consideración para la proporción entre experimentos que usa la identidad como matriz de varianzas-covarianzas de los que no.

Para poder contrastar los resultados del algoritmo propuesto, se aplicaran los siguientes métodos a la búsqueda de modelos de regresión (implementados en la función *regsubsets* del paquete *leaps* de *R* versión 3.5.3) :

- Backward
- Búsqueda exhaustiva
- Forward
- Reemplazo Secuencial

En cada ejecución se evaluaran los siguientes elementos:

Cuadro 2: Diccionario de datos de las simulaciones.

Titulo	Descripción
Semilla	Número de semilla
Intercepto	Número booleano; 1 := indica que el modelo simulado contiene intercepto, 0 := otro caso.
MD_Ortonormal	Número booleano; 1 := indica que las variables son ortogonales y estandarizadas, 0 := otro caso.
m	Número de variables del modelo "real".
SCRes	Suma de cuadrados de los residuales del modelo "real".
Metodo	Nombre del algoritmo de búsqueda.
Tiempo	Tiempo de ejecución en segundos.
$M(m)$	Suma de cuadrados de los residuales del modelo con (m) variables.
AIC	Menor AIC producido por el algoritmo de búsqueda.
M_AIC	Número de variables en la regresión con el mejor AIC.
BIC	Menor BIC producido por el algoritmo de búsqueda.
M_BIC	Número de variables en la regresión con el mejor BIC.
CP	Menor Cp de Mallows producido por el algoritmo de búsqueda.
M_CP	Número de variables en la regresión con el mejor Cp de Mallows.
AMEMIYA	Menor coeficiente del criterio propuesto en Amemiya, T. (1976) producido por el algoritmo de búsqueda.
M_AMEMIYA	Número de variables en la regresión con el mejor AMEMIYA.

Antes de mostrar los resultados se explicara la estructura del código para mayor claridad.

Simulaciones en R project

En la próxima página se desarrollan comentarios útiles para la comprensión del código de simulación y los resultados, la rutina completa se encuentran al final del apéndice de la tesis.

1. Lo primero es descargar los paquetes que se usaron en la programación, se realiza con la siguiente línea:

```
#Se instalan los paquetes necesarios
RNSGA2::instalar_paquetes()
```

2. Se crea una matriz para respaldar los datos de interés de cada simulación.

```
#analisis es un data frame que contiene las estadisticas de cada ejecucion.
analisis<-as.data.frame(matrix(nrow=1,ncol=67))
colnames(analisis)<-c("Semilla", "Metodo","Tiempo","Intercepto","MD_Ortonormal",
                      "M","SCRes","M_AIC","AIC","M_BIC","BIC","M_CP","CP",
                      "M_AMEMIYA","AMEMIYA", paste0("M",0:51))
```

3. Se inicializa la semilla de números aleatorios, el número de covariables, el tamaño de muestra y se declara un índice auxiliar (en ese orden).

```
#semilla que fija las simulaciones
semilla <-1
#numero de variables
n_var <-50
#numero de observaciones
tm <- 100
#indice que corre sobre las filas de la matriz de analisis
k<-1
```

4. Se declara una función que calcula el AIC, BIC, CP, AMEMIYA, SCE y el número de variables de los modelos que se obtienen de los algoritmos evaluados.

```
#se creara una funcion para crear una matrix con el numero de variables
#y cuatro criterios de seleccion de modelos: AIC, BIC, CP, el criterio
#de Amemiya y la suma de cuadrados de los residuales.
criterios<-function(modelos){...}
```

5. Dentro de un ciclo de 100 simulaciones:

```
while(semilla <= 100){...}
```

5.1. Se fija la semilla de números aleatorios.

```
 set.seed(semilla)
```

5.2. Se simulan datos por casos, con una matriz de diseño general u ortogonal y con o sin intercepto.

```
if((semilla %% 4)==1){...}
# 1: SE SIMULA UN MODELO SIN INTERCEPTO CON MATRIZ DE DISENO ORTOGONAL
if((semilla %% 4)==2){...}
# 2: SE SIMULA UN MODELO CON INTERCEPTO CON MATRIZ DE DISENO ORTOGONAL
if((semilla %% 4)==3){...}
 # 3: SE SIMULA UN MODELO SIN INTERCEPTO CON MATRIZ DE DISENO GENERAL
if((semilla %% 4)==0){...}
# 4: SE SIMULA UN MODELO CON INTERCEPTO CON MATRIZ DE DISENO GENERAL
```

Donde:

Lo primero es guardar el tipo de simulación.

```
# 1: SE SIMULA UN MODELO SIN INTERCEPTO CON MATRIZ DE DISENO ORTOGONAL
    analisis$Semilla[k] <- semilla
    analisis$Intercepto[k] <- FALSE
    analisis$MD_Ortonormal[k] <- TRUE
```

Se generan los parámetros de la regresión simulada (betas y soporte), la varianza de los residuales es igual a la unidad.

```
#se simulan los parametros del modelo
    adn <- c(0,rbinom(n_var,1,runif(1)))
    betas <- rnorm(n_var+1)*adn
```

Se construyen tanto la variable respuesta como las covariables, las ultimas de una normal centrada en el origen con una matriz de covarianzas “Sigma”.

```
#y generan los datos del mismo sin correlaciones entre covariables
datos <- RNSGA2::generar_datos(betas,adn,tm,Sigma = diag(n_var))
datos <-as.data.frame(datos)
datos$y <- datos$y - mean(datos$y)
```

5.3. Se calculan y respaldan los valores del modelo con todas (y sólo) las variables adecuadas.

```
#estimamos el modelo simulado
modelo <- RNSGA2::modelar(adn, datos)
analisis$M[k] <- sum(adn)
analisis$SCRes[k] <- sum(modelo$residuals^2)
rm(modelo)
```

5.4. Después se ejecuta el método de búsqueda.

```
analisis$Metodo[k]<-"Backward"
tiempos<-system.time(
  backward <- leaps::regsubsets(y~., data = datos,
                                method="backward",
                                nvmax = 20,
                                intercept=analisis$Intercepto[k],
                                really.big=TRUE)
)
backward <- summary(backward)$which

#calculando los modelos
modelos_backward <- apply(backward,1,
                          function(x){RNSGA2::modelar(
                            if(analisis$Intercepto[k]){x}else{c(0,x)},
                            datos)})
```

Se genera una lista de modelos de regresión lineal, donde cada elemento tiene un número de variables diferente a los demás.

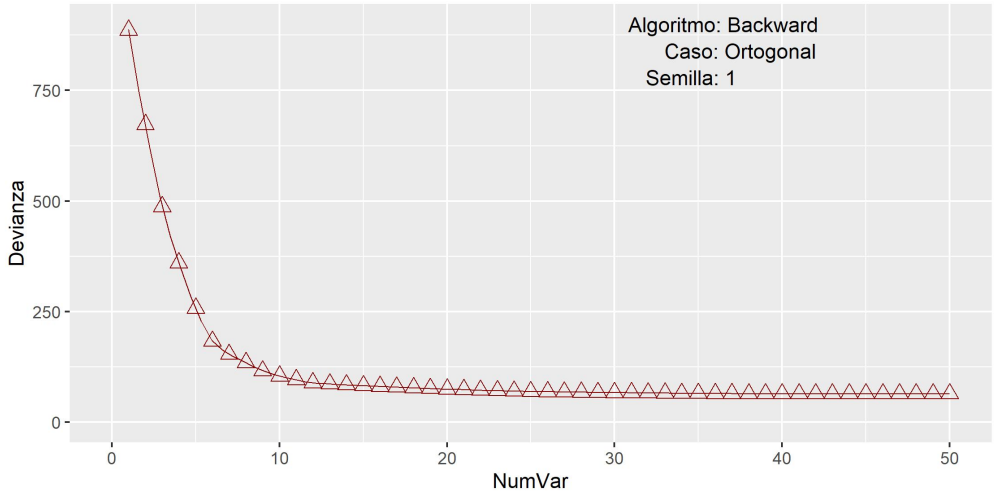

Figura 23: Gráfica de la lista de modelos que se pueden obtener mediante Backward con los datos de la primera simulación (caso ortogonal).

El paso anterior se repite para cada uno de los algoritmos de búsqueda de modelos, por lo que se obtendrán 5 listas. Así, para comparar los resultados de los métodos se calculan algunas estadísticas sobre las listas (véase el diccionario de datos).

5.5. Se guardan las estadísticas de cada lista de modelos. La tabla con los valores respaldados se encuentra dentro del paquete *RNSGA2* con el nombre "analisis" (sin acentos) .

```
#evaluando el tiempo de ejecucion
analisis$Tiempo[k] <- tiempos[1]

#se agregan las sumas de cuadrados de los residuales a la tabla de analisis
analisis[k,
         sapply(paste0("M",MatrizCriterios$NumVar),
                function(nombre){
                which(nombre==colnames(analisis))})] <- MatrizCriterios$SCE

#se agregan los valores del modelo seleccionado
analisis[k,
         sapply(colnames(MatrizCriterios[,-c(1,6)]),
                function(nombre){
                  which(colnames(analisis)==nombre)})
         ] <- apply(MatrizCriterios[,-c(1,6)], 2, min)

#y su numero de variables
analisis[k,
         sapply(c("M_AIC","M_BIC","M_CP","M_AMEMIYA"),
                function(nombre){
                  which(colnames(analisis)==nombre)})
         ] <- apply(MatrizCriterios[,-c(1,6)],
                    2,
                    function(valor){
                      MatrizCriterios$NumVar[which.min(valor)]
                    })

k<-k+1
analisis[k,c(1,4,5,7,8)]<-analisis[k-1,c(1,4,5,7,8)]
```

La estructura del código se repite para cada algoritmo.

Es importante mencionar que el análisis de resultados se realizó por casos, para respaldar las proposiciones del caso ortogonal.

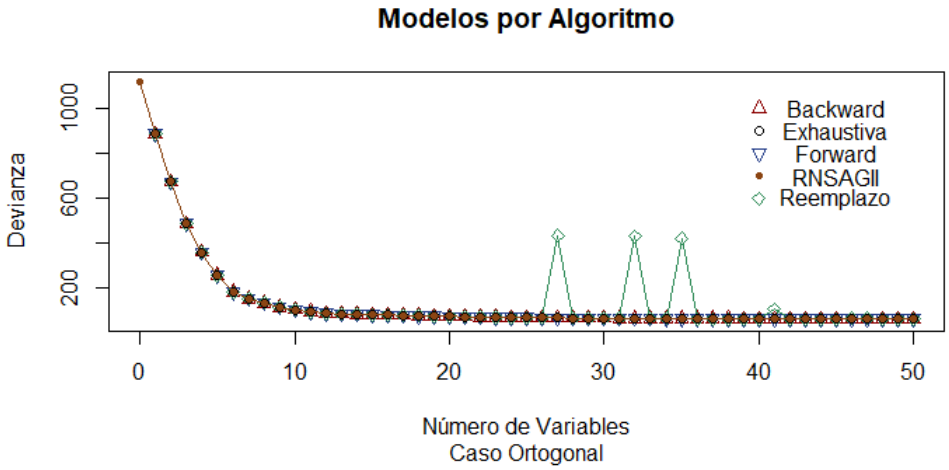

Figura 24: Cuadro comparativo de las listas de modelos de la primera simulación (caso ortogonal sin intercepto).

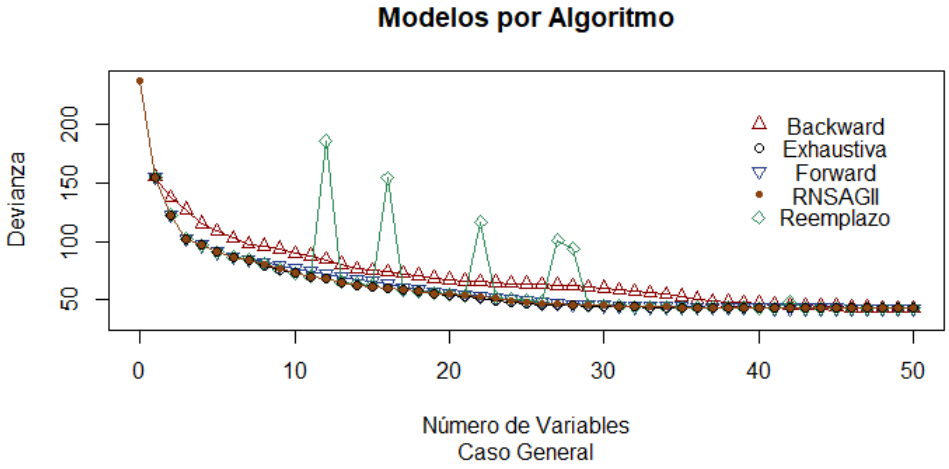

Figura 25: Cuadro comparativo de las listas de modelos de la tercera simulación (caso general sin intercepto).

Así, se repitió 100 veces el experimento y los resultados fueron los siguientes.

Resultados

En el cuadro 3 se encuentra la media de las diferencias de los valores que toman los criterios de selección (AIC, Amemiya, BIC, y Cp de Mallows) de los modelos elegidos de acuerdo con el AIC, Amemiya, BIC ó Cp de Mallows producidos por cada uno de los algoritmos de búsqueda, cuando $\mathbf{X}'\mathbf{X}$ es una matriz diagonal.

Por ejemplo, si se quisiera comparar el desempeño de Reemplazo secuencial contra el NSGA-II comparado por AIC, se calcularía lo siguiente:

$$\frac{\Sigma_{t=1}^{50} \text{mín}\{AIC(Reemplazo,k,t)|k=1,\ldots,50\} - \text{mín}\{AIC(NSGAII,k,t)|k=1,\ldots,50\}}{50}.$$

Donde $AIC(Reemplazo,k,t)$ es el AIC del modelo con k variables producido por el método de Reemplazo en la simulación número t y análogamente $AIC(NSGAII,k,t)$.

En la tabla se encuentra el valor 0.02382, que significa que en promedio el AIC producido por el método de Reemplazo Secuencial es 0.02382 más grande que el obtenido por NSGA-II, cuando $\mathbf{X'X}$ es una matriz diagonal. Análogamente, si se quisiera comparar la búsqueda exhaustiva contra el NSGA-II se obtendría el valor -0.00090, que implica que el AIC de la búsqueda exhaustiva es en promedio 0.00090 más chico que el del NSGA-II (comparando modelos con el mismo número de variables) en el caso $\mathbf{X'X}$ diagonal. Este

Cuadro 3: Diferencia de los valores de los criterios de selección por algoritmo (caso $X'X$ diagonal).

Caso: $\mathbf{X'X}$ diagonal						
Criterio	Algoritmo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo
AIC	Backward	0.00000	0.00000	0.00000	-0.00090	-0.02473
	Exhaustivo	0.00000	0.00000	0.00000	-0.00090	-0.02473
	Forward	0.00000	0.00000	0.00000	-0.00090	-0.02473
	NSGA-II	0.00090	0.00090	0.00090	0.00000	-0.02382
	Reemplazo	0.02473	0.02473	0.02473	0.02382	0.00000
Amemiya	Backward	0.00000	0.00000	0.00000	-0.00306	-0.00019
	Exhaustivo	0.00000	0.00000	0.00000	-0.00306	-0.00019
	Forward	0.00000	0.00000	0.00000	-0.00306	-0.00019
	NSGA-II	0.00306	0.00306	0.00306	0.00000	0.00287
	Reemplazo	0.00019	0.00019	0.00019	-0.00287	0.00000
BIC	Backward	0.00000	0.00000	0.00000	0.00000	-0.00284
	Exhaustivo	0.00000	0.00000	0.00000	0.00000	-0.00284
	Forward	0.00000	0.00000	0.00000	0.00000	-0.00284
	NSGA-II	0.00000	0.00000	0.00000	0.00000	-0.00284
	Reemplazo	0.00284	0.00284	0.00284	0.00284	0.00000
CP de Mallows	Backward	0.00000	0.00000	0.00000	0.00000	-0.01287
	Exhaustivo	0.00000	0.00000	0.00000	0.00000	-0.01287
	Forward	0.00000	0.00000	0.00000	0.00000	-0.01287
	NSGA-II	0.00000	0.00000	0.00000	0.00000	-0.01287
	Reemplazo	0.01287	0.01287	0.01287	0.01287	0.00000

cuadro muestra que cuando $\mathbf{X}'\mathbf{X}$ diagonal:

- En promedio, Backward, Forward y la búsqueda exhaustiva obtienen el mismo desempeño en todos los criterios de selección.
- Comparando el NSGA-II contra el Reemplazo Secuencial, se observa que NSGA-II es más eficiente que el Reemplazo en 3 de los 4 criterios (AIC, BIC y Cp de Mallow).

El cuadro 4 contiene la misma información que 3 para los casos donde $\mathbf{X}'\mathbf{X}$ no es diagonal. En la tabla anterior se puede observar que:

Cuadro 4: Diferencia de los valores de los criterios de selección por algoritmo (caso general).

Caso: $\mathbf{X}'\mathbf{X}$ no diagonal						
Criterio	Algoritmo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo
AIC	Backward	0.00000	10.86199	3.74865	9.84769	9.34917
	Exhaustivo	-10.86199	0.00000	-7.11334	-1.01429	-1.51281
	Forward	-3.74865	7.11334	0.00000	6.09904	5.60052
	NSGA-II	-9.84769	1.01429	-6.09904	0.00000	-0.49852
	Reemplazo	-9.34917	1.51281	-5.60052	0.49852	0.00000
Amemiya	Backward	0.00000	18.01484	12.87194	17.17827	16.63456
	Exhaustivo	-18.01484	0.00000	-5.14290	-0.83656	-1.38027
	Forward	-12.87194	5.14280	0.00000	4.30633	3.76262
	NSGA-II	-17.17828	0.83656	-4.30633	0.00000	-0.54371
	Reemplazo	-16.63456	1.38027	-3.76262	0.54371	0.00000
BIC	Backward	0.00000	17.72783	12.48193	16.78224	16.43139
	Exhaustivo	-17.72783	0.00000	-5.24589	-0.94558	-1.29643
	Forward	-12.48193	5.24589	0.00000	4.30031	3.94945
	NSGA-II	-16.78224	0.94558	-4.30031	0.00000	-0.35085
	Reemplazo	-16.43139	1.29643	-3.94945	0.35085	0.00000
CP de Mallows	Backward	0.00000	13.21897	7.56921	12.46936	11.89593
	Exhaustivo	-13.21897	0.00000	-5.64975	-0.74960	-1.32303
	Forward	-7.56921	5.64975	0.00000	4.90015	-4.32671
	NSGA-II	-12.46936	0.74960	-4.90015	0.00000	-0.57343
	Reemplazo	-11.89593	1.32303	-4.32671	0.57343	0.00000

- NSGA-II obtiene mejores resultados promedio que cualquier otro algoritmo de búsqueda diferente a la exhaustiva.
- Backward y Forward fueron los métodos peor evaluados en cualquiera de los cuatro criterios.

Las dos últimas tablas comparan los métodos de búsqueda a través de la diferencia promedio que producen en los criterios de selección. El próximo cuadro compara los algoritmos de búsqueda A y B a través de la frecuencia con la que A produce mejores (estrictamente

menores) AIC, Amemiya, BIC ó Cp de Mallows que B.

Por ejemplo, supóngase que se quiere comparar NSGA-II contra Forward usando el BIC, entonces se calcularía lo siguiente:

$$\Sigma_{t=1}^{50} \mathbb{1}_{[\text{mín}\{AIC(NSGAII,k,t)|k=1,...,50\}<\text{mín}\{AIC(Forward,k,t)|k=1,...,50\}]}$$

Como NSGA-II no generó un mejor AIC que Forward cuando $\mathbf{X}'\mathbf{X}$ es diagonal; toma el valor de 0. En cambio, cuando $\mathbf{X}'\mathbf{X}$ es no diagonal, toma el valor 37 porque NSGA-II genero 37 veces un mejor AIC que Forward.

Cuadro 5: Número de modelos preferibles por algoritmo de búsqueda y criterio de selección.

		$\mathbf{X}'\mathbf{X}$ diagonal					$\mathbf{X}'\mathbf{X}$ no diagonal				
Criterio	Algoritmo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo
AIC	Backward	0	0	0	1	3	0	0	22	1	3
	Exhaustivo	0	0	0	1	3	49	0	49	29	39
	Forward	0	0	0	1	3	28	0	0	1	1
	NSGA-II	0	0	0	0	3	48	0	49	0	29
	Reemplazo	0	0	0	1	0	47	0	48	15	0
Amemiya	Backward	0	0	0	1	1	0	0	20	2	2
	Exhaustivo	0	0	0	1	1	49	0	49	31	40
	Forward	0	0	0	1	1	30	0	0	1	1
	NSGA-II	0	0	0	0	1	47	0	49	0	29
	Reemplazo	0	0	0	1	0	48	0	48	14	0
BIC	Backward	0	0	0	0	1	0	0	10	0	1
	Exhaustivo	0	0	0	0	1	50	0	39	22	23
	Forward	0	0	0	0	1	40	0	0	4	2
	NSGA-II	0	0	0	0	1	50	0	37	0	14
	Reemplazo	0	0	0	0	0	49	0	36	15	0
CP de Mallows	Backward	0	0	0	0	3	0	0	6	1	1
	Exhaustivo	0	0	0	0	3	49	0	44	26	36
	Forward	0	0	0	0	3	44	0	0	1	0
	NSGA-II	0	0	0	0	3	48	0	43	0	25
	Reemplazo	0	0	0	0	0	49	0	40	10	0

En esta última tabla se observa que:

- Backward y Forward empatan con la búsqueda exhaustiva cuando $\mathbf{X}'\mathbf{X}$ es diagonal, en caso contrario, son los métodos peor evaluados en todos los criterios.
- NSGA-II obtiene mejores resultados en todos los criterios que cualquier otro algoritmo de búsqueda diferente a la exhaustiva si $\mathbf{X}'\mathbf{X}$ es no diagonal.

Las próximas dos tablas muestran las diferencias promedio de comparar las sumas de cuadrados de los residuales de modelos producidos por diferentes métodos de búsqueda

pero con las mismas variables.

Por ejemplo: Imagínese que se desea comparar la búsqueda exhaustiva contra NSGA-II usando la suma de cuadrados de los residuales (SCR), así se realizaría el siguiente calculo:

$$\Sigma_{t=1}^{50} \frac{\Sigma_{k=0}^{50} SCR(Exhaustiva,k,t) - SCR(NSGAII,k,t)}{50}/50.$$

Así, en el caso $\mathbf{X'X}$ diagonal, la búsqueda exhaustiva tiene modelos con una SCR menor (0,00722 en promedio) con respecto a NSGAG-II.

Cuadro 6: Diferencia entre la suma de cuadrados de los residuales de algoritmos de búsqueda fijando el número de variables (caso $X'X$ diagonal).

	Caso: $\mathbf{X'X}$ diagonal				
Algoritmo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo
Backward	0.00000	0.00000	0.00000	-0.00722	-0.06333
Exhaustivo	0.00000	0.00000	0.00000	-0.00722	-0.06333
Forward	0.00000	0.00000	0.00000	-0.00722	-0.06333
NSGA-II	0.00722	0.00722	0.00722	0.00000	-0.05569
Reemplazo	0.06333	0.06333	0.06333	0.05569	0.00000

Cuadro 7: Diferencia entre la suma de cuadrados de los residuales de algoritmos de búsqueda fijando el número de variables (caso general).

Cuadro comparativo de algoritmos por SCR.					
	Caso: $\mathbf{X'X}$ no diagonal				
Algoritmo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo
Backward	0.00000	0.88666	0.31333	0.72333	0.54541
Exhaustivo	-0.88666	0.00000	-0.85166	-0.61083	-0.61500
Forward	-0.31333	0.85166	0.00000	0.79541	0.58666
NSGA-II	-0.72333	0.61083	-0.79541	0.00000	-0.13500
Reemplazo	-0.54541	0.61500	-0.58666	0.13500	0.00000

En este último cuadro se observa que:

- Backward y Forward empatan con la búsqueda exhaustiva cuando $\mathbf{X'X}$ es diagonal, en caso contrario, son los métodos peor evaluados en todos los criterios.
- NSGA-II obtiene mejores resultados en todos los criterios que cualquier otro algoritmo de búsqueda diferente a la exhaustiva si $\mathbf{X'X}$ es no diagonal.

Ahora se compararan cada par de algoritmos (A, B) apoyándose en el concepto de Dominancia de Pareto, calculando la media de los porcentajes del número de modelos generados por B que son dominados por algún elemento de A.

Por ejemplo, si se quisiera comparar el Reemplazo Secuencial contra NSGA-II, entonces se calcularía lo siguiente:

$$\Sigma_{t=1}^{50} \frac{\Sigma_{k=1}^{50} \mathbb{1}_{(Modelo(Reemplazo,k,t) \text{ es dominado por algún modelo de } NSGAII \text{ en la simulación } t)}}{50} / 50.$$

Donde $Modelo(Reemplazo, k, t)$ es una función que regresa un vector (k, SCR) con el número de variables k y la suma de cuadrados de los residuales de la semilla número t. Así, el 0,666 % de los modelos generados por NSGA-II están dominados por algún modelo generado por Reemplazo Secuencial.

Cuadro 8: Cuadro comparativo de algoritmos por dominancia de Pareto (caso $X'X$ diagonal).

	$\mathbf{X'X}$ diagonal				
Algoritmo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo
Backward	0.000 %	0.000 %	0.000 %	0.708 %	6.333 %
Exhaustivo	0.000 %	0.000 %	0.000 %	0.708 %	6.333 %
Forward	0.000 %	0.000 %	0.000 %	0.708 %	6.333 %
NSGA-II	0.000 %	0.000 %	0.000 %	0.000 %	6.333 %
Reemplazo	0.000 %	0.000 %	0.000 %	0.666 %	9.791 %

Cuadro 9: Cuadro comparativo de algoritmos por dominancia de Pareto (caso general).

	$\mathbf{X'X}$ no diagonal				
Algoritmo	Backward	Exhaustivo	Forward	NSGA-II	Reemplazo
Backward	0.000 %	0.000 %	32.250 %	9.875 %	18.625 %
Exhaustivo	88.666 %	0.000 %	85.166 %	61.083 %	61.500 %
Forward	63.583 %	0.000 %	0.000 %	2.666 %	12.125 %
NSGA-II	82.208 %	0.000 %	82.208 %	0.000 %	41.666 %
Reemplazo	79.791 %	0.000 %	77.375 %	28.625 %	6.291 %

En los últimos dos cuadros se verifica que:

- Backward y Forward empatan con la búsqueda exhaustiva cuando $\mathbf{X'X}$ es diagonal en 0.000 %, es decir, no hay modelo que domine a los que generan Backward ó Forward.
- NSGA-II obtiene mejores resultados en todos los criterios que cualquier otro algoritmo de búsqueda diferente a la exhaustiva si $\mathbf{X'X}$ es no diagonal.
- Reemplazo secuencial es el único algoritmo de búsqueda que puede generar modelos que dominen a otros encontrados por sí mismo, esto sucede porque los busca de manera independientemente.

Ahora se explicarán los resultados obtenidos vía simulación mediante la teoría expuesta:

1. La proposición (14.1) muestra que un modelo es elegido por un "buen criterio" si y sólo si pertenece a la frontera de Pareto de la función propuesta en (55).

De acuerdo a lo demostrado anteriormente, AIC, Amemiya, BIC y Cp de Mallows son buenos criterios, y el modelo que seleccionan debe estar en la frontera de Pareto, por lo que no debería ser dominado por algún otro. Y en las simulaciones vemos que la búsqueda exhaustiva siempre obtuvo las mejores evaluaciones (por todos los criterios) y ninguno de sus modelos es dominado por ninguno de los 4 métodos en cualquier caso.

2. Si $\mathbf{X}'\mathbf{X}$ es diagonal, todas las variables están estandarizadas y normalizadas; variando el parámetro α de Backward y Forward, se calcula la frontera de Pareto de la función (55).

Como se observó, el desempeño de Backward y Forward fue equivalente al de la búsqueda exhaustiva cuando $\mathbf{X}'\mathbf{X}$ es una matriz diagonal.

3. Como consecuencia de la proposición anterior, el algoritmo NSGA-II adaptado, sólo puede superar los resultados de Backward o Forward cuando $\mathbf{X}'\mathbf{X}$ no es diagonal.

Como NSGA-II busca la frontera de Pareto, y como tanto Backward como Forward la obtienen en el caso particular mencionado, no es posible que NSGA-II mejore sus resultados. En realidad, cualquier otro método de búsqueda basado en "buenos criterios" de selección, dominancia de Pareto o regularización dispersa no convexa puede a lo más empatar sus resultados.

Sí $\mathbf{X}'\mathbf{X}$ no es diagonal, entonces la adaptación de NSGA-II, puede superar (y lo hace si tiene suficiente tiempo) los resultados de Backward o Forward.

Para terminar la evaluación de los algoritmos se compararan los tiempos medios de ejecución medidos en segundos.

Cuadro 10: Tiempos medios (en segundos) de ejecución de los algoritmos de búsqueda por casos.

Algoritmo	$\mathbf{X}'\mathbf{X}$ diagonal	$\mathbf{X}'\mathbf{X}$ no diagonal
Backward	0.00269	0.00541
Exhaustivo	1445.84692	7361.97666
Forward	0.00769	0.00291
NSGA-II	503.68500	2154.69375
Reemplazo	0.10923	0.14041

Es fácil notar que la búsqueda exhaustiva y NSGA-II son mucho más costosos que Backward, Forward ó el Reemplazo Secuencial, pero al mismo tiempo son los mejor evaluados por otras métricas, en consecuencia, el uso de estas técnicas debe considerar el tiempo de ejecución disponible. Algo interesante es notar que el reemplazo secuencial ofrece buenos resultados en relativamente poco tiempo.

Por último se mostrarán dos tablas interesantes de acuerdo con las proposiciones que se han mostrado.

Por la proposición (13.1), se sabe que si un modelo es resultado de estimar un modelo lineal con regularización dispersa no convexa, entonces ese modelo se encuentra en la frontera de Pareto de la función (55). Es decir, la frontera de Pareto contiene todos los modelos de regularización dispersa no convexa, pero ¿qué sucede con el enunciado inverso? Aquí no se demuestra la proposición inversa, pero en (13.2) se exhibe que cuando la frontera de Pareto de la función (55) es convexa, entonces todos los modelos que se encuentren en la misma, también se pueden obtener por regularización dispersa no convexa. Así, por la hipótesis de convexidad, usar un modelo en la frontera de Pareto es equivalente a hacer regularización dispersa no convexa.

Pero, ¿cuándo sucede que la frontera de Pareto sea convexa? En el texto no se da una respuesta, pero evaluando directamente la convexidad de la frontera de Pareto de cada una de las simulaciones, nos puede dar una idea. La tabla próxima muestra la frecuencia con la que la frontera sea convexa de acuerdo a los casos de la matriz $\mathbf{X}'\mathbf{X}$.

Cuadro 11: Convexidad de la frontera de Pareto de las simulaciones.

Casos	$\mathbf{X}'\mathbf{X}$ diagonal	$\mathbf{X}'\mathbf{X}$ no diagonal
Frontera convexa	50	0
Frontera no convexa	0	50

Lo anterior sugiere que la frontera de Pareto es convexa si y sólo si $\mathbf{X}'\mathbf{X}$ es diagonal, examinar ésta proposición es importante, pues de ser cierta, significaría que usar Backward o Forward (cuando $\mathbf{X}'\mathbf{X}$ es diagonal) es equivalente a hacer regularización dispersa no convexa. Una consecuencia de lo anterior es que no son válidos los estimadores, los intervalos de confianza y muchos de los resultados que ocupan el supuesto de considerar que la regresión cuenta con todas las variables correctas.

Como se ha visto, los modelos que son seleccionados por criterios como el AIC, Amemiya, BIC ó Cp de Mallows, se encuentran en la frontera de Pareto, entonces es interesante analizar la equivalencia que sugieren los datos pues en el caso adecuado, usar AIC, Amemiya, BIC, Cp de Mallos ó cualquier otro "buen criterio" es equivalente a estimar con regularización dispersa no convexa.

Conclusiones

Al inicio de este trabajo el objetivo era encontrar un caso particular donde el uso del NSGA-II tuviera una ventaja respecto de los algoritmos existentes, y tal suceso ocurre cuando las variables independientes no son ortogonales, pero en el caso contrarío ocurría que el algoritmo genético no superaba los resultados de los métodos clásicos de selección de variables (Backward y Forward), incluso tomando muchos recursos computacionales. Ese comportamiento se podría explicar como un problema de programación, como un inconveniente con función objetivo, ó simplemente como el hecho de que Backward y Forward ya generaban la frontera que busca el NSGA-II.

Cuando ya se hayan descartado las primeras dos opciones; sólo queda pensar en demostrar la última proposición, y esto se alcanzó como un corolario del primer teorema del artículo de Herlands, De-Arteaga, Neill y Dubrawski 2015, que habla de la relación entre la regresión dispersa no convexa y Lasso.

Además de lo anterior, se logran demostrar algunas relaciones entre los criterios de selección de modelo y la frontera de Pareto. Estos resultados se resumen en los siguientes puntos:

1. Se propone una definición de "buen criterio" y se demuestra que el AIC, Amemiya, BIC y Cp de Mallows son "buenos criterios".

2. Un modelo es elegido por un "buen criterio" si y sólo si pertenece a la frontera de Pareto de la función objetivo que se nombró como canónica.

   Como consecuencia, la frontera de Pareto de la función canónica permite encontrar los modelos elegidos por cualquier "buen criterio".

3. Cualquier modelo lineal con regularización dispersa no convexa se encuentra en la frontera de Pareto de la función canónica.

4. Cuando la frontera de Pareto de la función canónica es convexa se cumple la proposición inversa del punto anterior, todos los modelos en la frontera se pueden ver como un modelo con regularización dispersa no convexa.

   ¿Y sí la frontera de Pareto no es convexa?, en ese caso sólo tenemos asegurado que la frontera contiene todas las soluciones por regularización (dispersa no convexa) y por criterios de selección. Es decir, contiene modelos que con propiedades interesantes y

por lo tanto queda justificada la aproximación a estos mediante el uso del algoritmo NSGA-II.

5. Cuando las covariables son ortogonales, todos los modelos de la frontera de Pareto de la función canónica son modelos que resultan de el algoritmo de selección de variables Backward ó Forward para algún nivel de significancia (α) adecuado, se tiene que considerar que las variables independientes deben estar estandarizadas, normalizadas y sólo se consideran modelos sin el intercepto.

6. Como consecuencia de la proposición anterior, el algoritmo NSGA-II adaptado, sólo puede superar los resultados de Backward o Forward cuando las covariables no son ortogonales.

Con variables explicativas ortogonales cualquier método de búsqueda basado en "buenos criterios" de selección, dominancia de Pareto o regularización dispersa no convexa puede a lo más empatar los resultados de Backward o Forward.

7. NSGA-II tiene ventajas prácticas sobre otros métodos, por ejemplo:

- Paralelismo implícito.
- El criterio de paro puede ser definido en términos del tiempo.
- Es un algoritmo elitista, por lo que solo puede mejorar o mantener las soluciones entre generaciones. Particularmente útil si se inicializa la población con algún método de optimización.
- Se puede agregar restricciones al conjunto de búsqueda, asignando valores máximos a los modelos que no cumplan las restricciones. Por ejemplo: modelos con intercepto, modelos que no rechacen la normalidad de los residuales, etc.

Apéndice

En esta parte del apéndice se encuentran cada una de las funciones que componen el código del algoritmo *RNSGAII* en orden alfabético:

cruzar

```
#' @title Operador de cruzamiento
#' @description A partir de dos vectores de largo n con entradas binarias,
#' crea una matriz de entradas binarias de dimensiones 2 x n a partir de los mismos.
#' @param adn vector de entradas binarias: representa un modelo de regresion lineal.
#' Donde, si en la posicion p, el vector tiene 0:= la variable p no se encuentra en
#' la regresion, 1:= la variable p se encuentra en la regresion
#' y p=1 representa la contante al origen.
#' @return Vector de 2n con entradas en {0,1}.
#' @examples
#' library(RNSGA2)
#' (adn1<-rbinom(10,1,0.5))
#' (adn2<-rbinom(10,1,0.5))
#' (cruzar(adn1,adn2))
#'
cruzar<-function(adn1,adn2){
  #esta funcion regresa una matriz con la cruza de dos pobladores
  n_var<-length(adn1)-1
  adn1<-as.numeric(adn1)
  adn2<-as.numeric(adn2)
  #se genera un numero entero entre 2 y el numero de variables
  #con probabilidad uniforme
  bloque<-as.integer(runif(1,2,(n_var+1)))
  #los descendientes se representaran por la matriz "hijo"
  hijo<-matrix(nrow=2, ncol=(n_var+1))
  hijo[1,]<-c(adn1[1:bloque],adn2[(bloque+1):(n_var+1)])
  hijo[2,]<-c(adn2[1:bloque],adn1[(bloque+1):(n_var+1)])
  return(c(hijo[1,],hijo[2,]))
}
```

extincion

```
#' @title Operador de reemplazo
#' @description Crea un vector de enteros que representa a los pobladores que pasan
#' a la siguiente generacion.
#' @param cd Lista que contiene la crowdistance de cada elemento de la poblacion
#' agrupados por frontera.
#' @param n_mdls Entero positivo: que representa el numero de pobladores.
#' @param ranking Ranking que contiene el numero de poblador agrupados por frontera.
#' @return Vector de enteros de tamaqo n_mdls.
extincion<-function(n_mdls,ranking,cd){
  #Esta funcion selecciona a los pobladores que sobreviven
  #i es un indice que corre sobre el numero de fronteras
  i<-1
  #sobrevivientes es una matriz que contiene a los pobladores
  #de la proxima generacion
```

```
  sobrevivientes<-c()
  while(length(sobrevivientes)<n_mdls){
    #si al agregar los elementos de la frontera i no se rebasa
    #el nC:mero de pobladores, entonces sobrevive toda la frontera
    if((length(ranking[[i]])+length(sobrevivientes))<=n_mdls){
      sobrevivientes<-c(sobrevivientes,ranking[[i]])
    }else{
      #en caso contrario, se seleccionan a los elementos de mas diversidad
      #aux contiene la crowdistan de los elementos de la frontera
      aux<-cd[ranking[[i]]]
      #aux ordena de mayor a menor crowdistan
      aux<-order(aux, decreasing=TRUE)
      #se seleccionan los mas diversos
      aux<-aux[1:(n_mdls-length(sobrevivientes))]
      #se agregan a lo sobrevivientes
      sobrevivientes<-c(sobrevivientes,ranking[[i]][aux])
    }
    i<-i+1
  }
  return(sobrevivientes)
}
```

extraer_adn

```
#' @title Codificacion de modelos de regresion
#' @description Esta funcion codifica el soporte de una regresion como un
#' vector binario.
#' @param modelo Objeto de tipo lm.
#' @param nombres Vector de textos que contiene el nombre de las variables.
#' @return Vector con entradas en {0,1}.
#' @examples
#' library(RNSGA2)
#' vmodl<-c(1,rbinom(14,1,0.5))
#' betas<-rnorm(15)*vmodl
#' tam_mues<-30
#' datos<-generar_datos(betas, vmodl, tam_mues)
#' datos<-as.data.frame(datos)
#' modelo<-lm(y ~ var1 + var2 + var10, data= datos)
#' (extraer_adn(modelo,colnames(datos)))
#'
extraer_adn<-function(modelo,nombres=nombres){
  #esta funcion codifica la variables que tiene el modelo
  #como una cadena binaria. Donde 1 si se incluje la j-esima varible
  #y 0 en otro caso.
  #el primer espacio se reserva para la ordenada al origen
  adn<-as.numeric(nombres %in% all.names(formula(modelo)))
  if (is.na(modelo$coefficients["(Intercept)"])[[1]]){
    adn[1]<-0
  }else{
    adn[1]<-1
  }
  return(adn)
}
```

formular

```
#' @title Decodificacion de modelos de regresion
#' @description Esta funcion decadifica un vector binario (el soporte de una
#' regresion) como un objeto de la clase 'formula'.
#' @param adn Vector binario.
#' @param nombres Vector de textos que contiene el nombre de las variables.
#' @return Objeto de la clase formula.
#' @examples
#' library(RNSGA2)
```

```
#' vmodl<-c(1,rbinom(14,1,0.5))
#' betas<-rnorm(15)*vmodl
#' tam_mues<-30
#' datos<-generar_datos(betas, vmodl, tam_mues)
#' datos<-as.data.frame(datos)
#' formular(vmodl,colnames(datos))
#'
formular<-function(adn, nombres=nombres){
  #esta funcion de codifica una cadena binaria a una expresion
  #de tipo formula (ej. y~x)
  if((sum(adn)==0)|((adn[1]==1)&(sum(adn)==1))){

    if(sum(adn)==1){
      #si el modelo solo tiene constante regresa algo similar a y~1
      formula<-paste0(nombres[1],"~1",collapse="")
    }else{
      #si el modelo no tiene variables ni constante regresa algo similar a y~0
      formula<-paste0(nombres[1],"~0",collapse="")}
  }else{
    #en caso contrario devolvera una expresion del tipo y~x1+x2+...+xn
    #(si no tiene constante y~0+x1+x2+...+xn)
    arn<-order(adn,decreasing=TRUE)[1:sum(adn)]

    if(arn[1]==1){
      arn<-arn[-1]
      vrbls<- paste(nombres[arn],collapse="␣+␣")
    }else{
      vrbls<- paste(c(0,nombres[arn]),collapse="␣+␣")
    }
    formula<-as.formula(paste(nombres[1],"~",vrbls,collapse=""))
  }
  return(formula)
}
```

generar_datos

```
#' @title Generacion de datos para simulacion
#' @description Genera una matriz de tam_mues observaciones simuladas
#' de una recta de parametros 'betas'.
#' @param betas Vector real que representa los parametros de la regresion,
# incluyendo la contante en primer termino.
#' @param ds ds es la desviacion estandar de los residuales.
#' @param tam_mues Entero positivo, representa el numero de observaciones
#' del plano de regresion.
#' @param Sigma Matriz de Varianzas-Covarianzas de las covariables.
#' @param vmodl Vector binario que representa el soporte de la regresion, es
#' decir 0:= beta_i = 0 contra 1:= beta_i != 0.
#' @return Matriz de tam_mues x n_var + 1 con entradas en los reales.
#' @examples
#' library(RNSGA2)
#' (vmodl<-c(1,rbinom(14,1,0.5)))
#' (betas<-rnorm(15)*vmodl)
#' tam_mues<-10
#' generar_datos(betas, vmodl, tam_mues)
generar_datos<-function(betas,vmodl, tam_mues,Sigma = diag(length(betas)-1),ds=1){
  #esta funcion simula datos de una regresion
  #betas es el vector de parametros
  #vmodl es la codificacion de la regresion
  #tam_mues es el numero de observaciones que se calculan con la regresion
  #ds es la desviacion estandar de los residuales
  n_var<-length(betas)-1
  #se obtiene la matriz de disenio a partir de una normal multivariada
  mu <- rep(0,n_var)
  x1<-MASS::mvrnorm(n=tam_mues, mu=mu, Sigma=Sigma, empirical = TRUE)
  x1<-matrix(x1,nrow=tam_mues)
  colnames(x1)<-paste0("var",1:n_var)
```

```
  #se simulan los residuales de una normal estandar
  ruido<-rnorm(tam_mues,0,ds)
  #se calcula los valores esperados de la regresion
  Z<-betas[1]+apply(t(betas[2:(n_var+1)]*t(x1)),1,sum)
  y<-Z+ruido
  rm(ruido)
  #se correlacionan las variables que no incluya el modelo
  libres<-which(vmodl==0)
  x<-cbind(y,x1)
  nombres<<-colnames(x)

  lista1<<-list(upper = paste("~",paste(nombres[2:(1+n_var)],
                        collapse="+")), lower = ~1)

  if (all(is.na(x))){
    x<-generar_datos(betas,vmodl, tam_mues,Sigma)
  }
  return(x)
}
```

iniciar_poblacion

```
#' @title Inicializa una poblacion de NSGAII adaptado
#' @description Crea una matriz de n_mdls x n_var que representa
#' a la poblacion inicial del algoritmo genetico, generada del soporte
#' de las regresiones tipo Lasso y modificaciones aleatorias.
#' Donde n_mdls es el numero de individuos contenidos por la
#' poblacion e 'i' es un indice que corre sobre las filas de la matriz
#' (de 1 hasta n_mdls).
#' @param poblacion Matriz de n x n_var con entradas en {0,1},
#' con vectores que se desean incluir en la poblacion inicial.
#' @param n_mdls Entero positivo: que representa el numero de pobladores.
#' @param n_var Entero positivo: representa el numero de variables consideradas
#' para el analisis de regresion.
#' @return Matriz de n_mdls x n_var con entradas en {0,1}.
#' @examples
#' library(RNSGA2)
#' vmodl<-c(1,rbinom(14,1,0.5))
#' betas<-rnorm(15)*vmodl
#' tam_mues<-10
#' datos<-generar_datos(betas, vmodl, tam_mues)
#' iniciar_p(5,10,datos)
#'
iniciar_poblacion<-function(n_mdls, n_var=(dim(datos)[2]-1),
                                  poblacion=c(), datos=datos){
  #Esta funcion inicializa la primera generacion del lagoritmo genetico
  #n_mdls es el numero de pobladores
  #n_var es el numero de columnas menos una que cuenta cada poblador
  #poblacion es una matriz y un parametro opcional, por si se quiere
  #inicializar la poblacion manualmente

  #se verifica que no se sobrepase el numero de modelos
  lim_m<-min(n_mdls,exp((1+n_var)*log(2)))
  n_mdls<<-lim_m
  num<-1 #la variable "num" es un contador del numero de pobladores
  #Si no se ingresa una poblacion inicial se sustituye
  if(class(poblacion)=="NULL"){
    solucion_lasso <- glmnet::glmnet(scale(datos[,-c(1)]),
                        datos[,c(1)]-mean(datos[,c(1)]))
    solucion_lasso $beta<-t(as.matrix(solucion_lasso$beta))
    poblacion<-t(apply(solucion_lasso$beta,1,function(fila){
      sapply(fila, function(a){if(a==0){0}else{1}})
    }))
    poblacion<-rbind(poblacion,rep(1,dim(poblacion)[2]))
    poblacion<-rbind(poblacion,rep(0,dim(poblacion)[2]))
    poblacion<-unique(poblacion)
```

```
    poblacion<-rbind(
      cbind(rep(0,dim(poblacion)[1]),poblacion),
      cbind(rep(1,dim(poblacion)[1]),poblacion)
      )

  }else{
    dimensiones<-dim(poblacion)
    if(dimensiones[2]!=(1+n_var)){
      print("Error en el numero de columnas de la poblacion inicial")
      solucion_lasso <- glmnet::glmnet(scale(datos[,-c(1)]),
                                       datos[,c(1)]-mean(datos[,c(1)]))
      solucion_lasso $beta<-t(as.matrix(solucion_lasso$beta))
      poblacion<-t(apply(solucion_lasso$beta,1,function(fila){
        sapply(fila, function(a){if(a==0){0}else{1}})
      }))
      poblacion<-rbind(poblacion,rep(1,dim(poblacion)[2]))
      poblacion<-rbind(poblacion,rep(0,dim(poblacion)[2]))
      poblacion<-unique(poblacion)
      poblacion<-rbind(
        cbind(rep(0,dim(poblacion)[1]),poblacion),
        cbind(rep(1,dim(poblacion)[1]),poblacion)
      )
    }else{
      #se quitan los repetidos
      poblacion<-unique(poblacion)
      dimensiones<-dim(poblacion)
      if(dimensiones[1]>n_mdls){poblacion<-poblacion[1:n_mdls,]}
      num<-dimensiones[1]
    }
    rm(dimensiones)
  }
  #hasta tener el numero deseado de modelos generados aleatoriamente
  while(num<lim_m){
    i<-as.integer(runif(1,1,dim(poblacion)[1]+1))
    poblacion<-rbind(poblacion,mutar(poblacion[i,],1,1/dim(poblacion)[2]))
    #se verifica que cada modelo sea diferente
    poblacion<-unique(poblacion)
    num<-dim(poblacion)[1]
  }
  colnames(poblacion)<-c("Intercepto", colnames(datos)[-c(1)])
  rownames(poblacion)<-rep("",dim(poblacion)[1])
  return(poblacion[1:n_mdls,])
}
```

instalar_paquetes

```
#' @title Instala los paquetes usados por RNSGA2
#' @description Instala los paquetes necesarios (en caso de que no se
#' encuentren instalados) para poder ejecutar tanto el algoritmo principal
#' (RNSGAII), ejemplos y simulaciones:
#'
#' 'doParallel', 'foreach', 'glmnet', 'lars', 'leaps', 'lmtest', 'MASS',
#' 'Matrix', 'mipfp', 'nsga2R', 'olsrr', 'parallel' y 'stats'
#'
#' @examples
#' library(RNSGA2)
#' Instalar()
#'
instalar_paquetes<-function(){
  paquetes<-c("doParallel", "parallel", "nsga2R", "MASS", "stats", "foreach",
              "lmtest", "leaps","lars","olsrr","glmnet","Matrix","mipfp")
  paquetes <- paquetes[!(paquetes %in% installed.packages()[,"Package"])]
  if(length(paquetes)) install.packages(paquetes)
}
```

modelar

```
#' @title Decodificacion de modelos de regresion
#' @description Esta funcion decodifica un vector binario (el soporte
#' de una regresion) como un objeto de la clase 'lm'.
#' @param adn Vector binario que representa el soporte de una regresion.
#' @param datos Data frame con entradas reales, donde las columnas
#' representan variables.
#' @return Objeto de la clase lm.
#' @examples
#' library(RNSGA2)
#' vmodl<-c(1,rbinom(14,1,0.5))
#' betas<-rnorm(15)*vmodl
#' tam_mues<-30
#' datos<-generar_datos(betas, vmodl, tam_mues)
#' datos<-as.data.frame(datos)
#' (modelar(vmodl,datos))
#'
modelar<-function(adn,datos){
  nombres<-colnames(datos)
  if(sum(adn)==0){
    formula<-as.formula(paste0(nombres[1],"~0",collapse = ""))
  }else{
    if((sum(adn)==1)&&(adn[1]==1)){
      formula<-as.formula(paste0(nombres[1],"~1",collapse = ""))
    }else{
      formula<-formular(adn,nombres)
    }
  }
  modelo_adn<-stats::lm(formula,data=datos)
}
```

mutar

```
#' @title Operador de mutacion
#' @description Esta funcion realiza cambios aleatorios en el vector 'adn'
#' que no modifican sus dimensiones, unicamente sus entradas (o genes), con
#' una probabilidad 'psm' de seleccion y una probabilidad 'pmg' de cambio de
#' 0 a 1, o de 1 a 0.
#' @param adn vector de dimension 'n' de entradas binarias: que representa
#' al adn que va mutar.
#' @param pmg numero real entre 0 y 1: probabilidad 'pmg' de cambio de 0 a
#' 1, o de 1 a 0.
#' @param psm numero real entre 0 y 1: probabilidad de seleccion de un gen
#' para sortear la mutacion.
#' @return Vector con 'n' entradas binarias.
#' @examples
#' library(RNSGA2)
#' set.seed(0)
#' psm<-0.5
#' pmg<-1
#' (adn<-rbinom(15,1,.5))
#' mutar(adn,psm,pmg)
#'
mutar<-function(adn,psm,pmg){
  adn<-as.vector(adn,"logical")
  if(rbinom(1,1,psm)==1){
    mutaciones<-which(rbinom(length(adn),1,pmg)==1)
    if(length(mutaciones)>0){
      x<-sapply(adn[mutaciones],function(g){if(g==FALSE){TRUE}else{FALSE}})
      adn[mutaciones]<-x}
  }
  adn<-as.numeric(adn)
  return(adn)
}
```

panel

```
#' @title Grafica de los valores objetivos
#' @description Esta funcion grafica los valores objetivos de una poblacion.
panel <- function(x, ...){
  usr <- par("usr"); on.exit(par(usr))
  #se define la region donde se graficara
  par(usr = c(usr[1:2], 0, 1.5) )
  histograma <- hist(x, plot = FALSE)
  saltos <- histograma$breaks;
  Num <- length(saltos)
  y <- histograma$counts; y <- y/max(y)
  rect(saltos[-Num], 0, saltos[-1], y, col="deepskyblue3", ...)
  #para dibujar los histogramas
}
```

rangos

```
#' @title Cambio de lista a vector los rangos de la poblacion
#' @description Esta funcion transforma el ranking (que es un objeto tipo
#' lista) en un vector de enteros que representa el numero de frontera en
#' la cual se encuantra cada poblador (en el mismo orden).
#' @param objetivos Matriz que contiene la valuacion de los pobladores.
#' @param ranking Ranking que contiene el numero de poblador agrupados
#' por frontera.
#' @return Vector de enteros con 'n_mdls' numero de entradas.
#'
rangos<-function(objetivos,ranking){
  #esta funcion transforma el ranking (que es un objeto tipo lista)
  #en un vector de enteros que representa el numero de frontera
  #en la cual se encuantra cada poblador (en el mismo orden)
  #se inicializa el rango de todos los pobladores en 0
  rnkIndex <- integer(dim(objetivos)[1])
  #se asician los pobladores a los rangos
  i <- 1
  while (i <= length(ranking)) {
    rnkIndex[ranking[[i]]] <- i
    i <- i + 1
  }
  return(rnkIndex)
}
```

reemplazo_secuencial

```
#' @title Reemplazo Secuencial
#' @description Optimiza un vector binario que representa un modelo
#' de regresion. Este algoritmo se usara para hibridar al NSGA-II adaptado.
#' @param adn vector de entradas binarias: representa un modelo
#' de regresion lineal.
#' @param cl Un objeto de la clase c("SOCKcluster", "cluster").
#' @param datos Matriz de datos: donde la primera columna contiene
#' la variable dependiente.
#' @return Vector binario de n_var + 1 con entradas en {0,1}.
reemplazo_secuencial<-function(adn,
                datos,
                cl = parallel::makeCluster(parallel::detectCores()-1)){
  parametros=c("modelar","datos","formular")
  cambio<-TRUE
  while (cambio) {
    cambio<-FALSE
    for(i in c(which(adn[-c(1)]==1)+1)){
      SCE<-sum(RNSGA2::modelar(adn,datos)$residuals^2)
      m0<-sum(adn)
```

```
      vecinos<-t(sapply(
        which(adn==0),
        function(j){
          vecino<-adn
          vecino[i]<-0
          vecino[j]<-1
          return(vecino)
        }))
      SCEs<-foreach(i=c(1:dim(vecinos)[1]),
         .combine=cbind,
         .export=parametros) %dopar% {
                tryCatch(sum(modelar(vecinos[i,],datos)$residuals^2),
                         error = function(e){Inf})      }
      if(min(SCEs)<SCE){
        cambio<-TRUE
        adn<-vecinos[which.min(SCEs),]
      }
    }
  }
  adn
}
```

RNSGAII

```
#' @title NSGAII adaptado para el uso en problemas de regresion lineal
#' @description Algoritmo de optimizacion multiobjetivo adaptado al
#' problema de regresion lineal multiple.
#' @param cl Un objeto de la clase c("SOCKcluster", "cluster").
#' @param datos Data frame con entradas reales, donde las columnas
#' representan variables.
#' @param funcion.fit Nombre de la funcion objetivo.
#' @param hibridar Entero, que representa la ciclicidad de hibridacion.
#' @param indxtorneo Entero positivo menor o igual a n_mdls, que
#' representa el numero de participantes en los torneos de seleccion.
#' @param iteraciones Numero de iteraciones del algoritmo.
#' @param parametros Vector con cadenas de texto que representan
#' objetos necesarios para la valuacion de la funcion objetivo.
#' @param poblacion_i Matriz con entradas binarias que representa
#' individuos que se desea que aparezcan en la poblacion inicial.
#' @param pmg Probabilidad de mutacion.
#' @param psm Probabilidad de seleccion de mutacion.
#' @param n_mdls Entero positivo: que representa el numero de pobladores.
#' @param Tparo1 Numero de segundos maximos que puede durar la ejecucion.
#' @param Tparo2 Numero de generaciones en las que no ha ocurrido una mejora
#' en la primera frontera y se considera que se ha alcanzado la Frontera de Pareto.
#' @return Objeto de la clase 'list' que contiene la frontera de Pareto de los
#' modelos de regresion lineal, la valuacion del la funcion objetivo y el historico
#' de  todos los elementos evaluados durante la ejecucion.
#' @examples
#' library(RNSGA2)
#' vmodl<-c(1,rbinom(14,1,0.5))
#' betas<-rnorm(15)*vmodl
#' tam_mues<-30
#' datos<-generar_datos(betas, vmodl, tam_mues)
#' datos<-as.data.frame(datos)
#' RNSGAII(datos,iteraciones=50)
#'
RNSGAII<-function(datos,
            cl = parallel::makeCluster(parallel::detectCores()-1),
            funcion.fit="valuar",
            hibridar=Inf,
            indxtorneo=max(as.integer(.1*n_mdls),2),
            iteraciones=1000,
            parametros=c("modelar","datos","formular"),
            poblacion_i=c(),
            pmg=1/dim(datos)[2],
```

```
          psm=.05,
          n_mdls=2*dim(datos)[2],
          Tparo1=Inf,
          Tparo2=100){
tiempo<- proc.time()[3]
print("#####################################################")
print("#␣␣␣␣␣␣␣␣␣␣␣␣␣␣INICIANDO␣LA␣EJECUCION␣␣␣␣␣␣␣␣␣␣␣␣␣␣#")
print("#####################################################")
#se nombrar algunas variables auxiliares para la ejecucion del codigo
#el numero de columnas es un dato que se necesita frecuentemente y se
#guarda en una variable global (numcol)
numcol<-dim(datos)[2]-1
#se guardan los nombres de los operadores de cruzamiento y mutacion
op.cruzamiento<-"cruzar"
op.mutacion<-"mutar"
n_var<-dim(datos)[2]-1
#parametros contiene informacion de las variables del ambiente global
#que se deben pasar a cada cluster de procesamiento paralelo
parametros<-c(parametros, op.cruzamiento, op.mutacion,
              funcion.fit, "poblacion", "psm", "pmg")
dimensiones <- dim(datos)
#se inicializa la poblacion
poblacion<-iniciar_poblacion(n_mdls,numcol,poblacion_i,datos)
#se observa el numero de entradas de la funcion objetivo
n_funciones<-length(do.call(eval(as.symbol(funcion.fit)),
                                      list(poblacion[1,],datos)))
#se llama la libreria "foreach" para poder ejecutar el codigo en paralelo
library(foreach)
#se guarda el ambiente para poder pasarlo a las ejecuciones en paralelo
ambiente<-environment()
#se registra el numero de clusters
doParallel::registerDoParallel(cl)
#se pasan los parametros
parallel::clusterExport(cl, parametros, envir=ambiente)
#se evalua cada elemento de la poblacion
  objetivos<-foreach(i=c(1:dim(poblacion)[1]),
                     .combine=cbind, .export=parametros) %dopar% {
    do.call(eval(as.symbol(funcion.fit)),list(poblacion[i,],datos))
  }


objetivos<-as.data.frame(t(objetivos))
#la matriz muestra guarda todos los elementos evaluados durante la ejecucion
muestra<-foreach(i=c(1:dim(poblacion)[1]),
              .combine=cbind, .export=parametros) %dopar% {
  paste0(poblacion[i,],collapse = "")
}
muestra<-as.data.frame(t(muestra))
#renombrando columnas
colnames(muestra)<-"Poblador"
muestra<-cbind(muestra,objetivos,0)
colnames(muestra)[dim(muestra)[2]]<-"Generacion"
#se grafica para poder mostrar el progreso durante la ejecucion
tryCatch(
  if(min(objetivos[,1])!=Inf){pairs(objetivos,panel=panel.smooth,  cex = 1,
                                    pch = 19, bg="blue", diag.panel=panel,
                                    cex.labels = 1, font.labels=1)},
                                    error=function(x){0})

#se inicializa el contador del numero de generaciones
generacion<-1
#calculando el numero de frontera para cada poblador
ranking<-nsga2R::fastNonDominatedSorting(objetivos)
objRange <- apply(objetivos, 2, max) -   apply(objetivos, 2, min)
rnkIndex <- rangos(objetivos,ranking)
objRange<-as.numeric(objRange)

#los modelos sustituidos en las ultimas Tparo2 generaciones
```

```
sustituciones<-rep(Inf,Tparo2)
oldw <- getOption("warn")
options(warn = -1)

while(
  tryCatch(
    (generacion<=iteraciones)&(sum(sustituciones)>=1),
    error=function(e){TRUE}
    )
  ){
  print("#####################################################")
  print(paste0("Generacion␣", generacion, collapse = "␣"))
  for(f in 1:length(ranking)){
    print(paste0("Pobladores␣en␣la␣frontera␣",f, ":␣", length(ranking[[f]]) ))
  }
  #guardando la primera frontera para calcular el numero de nuevos modelos
  f1_anterior<-ranking[[1]]
 #calculando crow distance
  cd<-nsga2R::crowdingDist4frnt(cbind(poblacion,rnkIndex),ranking,objRange)

  #seleccionando los elementos para cruzamiento
  poblador2<-nsga2R::tournamentSelection(cbind(cbind(poblacion,rnkIndex),
                      apply(cd,1,sum)),n_mdls,indxtorneo)[,1:(1+n_var)]


  #creando a la  nueva generacion de pobladores
  ngeneracion<-foreach(i=c(1:n_mdls),
                       .combine=cbind, .export=parametros) %dopar% {
    cruzar(poblacion[i,],poblador2[i,])
  }
  ngeneracion<-t(ngeneracion)
  ngeneracion<-rbind(ngeneracion[,1:(n_var+1)],
              ngeneracion[,(n_var+2):(2*(n_var+1))])
  rm(poblador2)
  #mutando la nueva generacion
  ngeneracion<-foreach(i=c(1:dim(ngeneracion)[1]),
                .combine=cbind, .export=parametros) %dopar% {
    mutar(as.integer(ngeneracion[i,]),psm,pmg)
  }

  #se eliminan elementos repetidos
  ngeneracion<-t(ngeneracion)
  #se hibrida el algoritmo
  if(hibridar!=Inf){
    if(generacion%%hibridar==0){
      for(i in 1:n_mdls){
        print(paste0("Optimizando␣el␣poblador␣numero:␣",i))
        optimizado <- reemplazo_secuencial(as.numeric(poblacion[i,]),datos, cl)
        ngeneracion<-rbind(ngeneracion,optimizado)
      }
    }
  }

  ngeneracion<-unique(ngeneracion)
  #pobladores es una variable auxiliar para respaldar la nueva generacion en "muestra"
  pobladores<-foreach(i=c(1:dim(ngeneracion)[1]),
              .combine=cbind, .export=parametros) %dopar% {
    paste0(ngeneracion[i,],collapse = "")
  }
  pobladores<-t(pobladores)
  #se eliminan repetidos
  repetidos<-foreach(i=c(1:dim(ngeneracion)[1]),
                .combine=cbind, .export=c("muestra","pobladores")) %dopar% {
    x<-which(muestra[,1]==pobladores[i,])[1]
    if(is.na(x)){x<-0}
    if(length(x)==0){x<-0}
    #if(is.nan(x)){x<-0}
    x
```

```
}
ngeneracion<-ngeneracion[which(repetidos==0),]
#evaluando la nueva generacion
if(is.null(dim(ngeneracion))!=TRUE){
  if(dim(ngeneracion)[1]>0){
    obj1<-foreach(i=c(1:dim(ngeneracion)[1]),
                  .combine=cbind, .export=parametros) %dopar% {
      do.call(eval(as.symbol(funcion.fit)),list(ngeneracion[i,],datos))
      }
    obj1<-t(obj1)
    poblacion<-rbind(poblacion,ngeneracion)
    objetivos<-rbind(objetivos,obj1)

    pobladores<-pobladores[which(repetidos==0)]
    pobladores<-as.data.frame(pobladores)
    colnames(pobladores)<-"Poblador"
    pobladores<-cbind(pobladores,obj1,generacion)
    colnames(pobladores)[dim(pobladores)[2]]<-"Generacion"
    muestra<-rbind(muestra,pobladores)
  }
}
rm(ngeneracion, pobladores)
#Calculando en numero de frontera de cada poblador
ranking<-nsga2R::fastNonDominatedSorting(objetivos)
objRange<- apply(objetivos, 2, max) -   apply(objetivos, 2, min)
objRange<-as.numeric(objRange)
rnkIndex <- rangos(objetivos,ranking)
#calculando el numero de nuevos modelos en la primera frontera
sustituciones[1:(Tparo2-1)]<-sustituciones[2:Tparo2]
sustituciones[Tparo2]<-length(which(!(ranking[[1]] %in% f1_anterior)))
#calculando la crow distance
cd<-nsga2R::crowdingDist4frnt(cbind(poblacion,rnkIndex),ranking,objRange)
#seleccionando los elementos que pasaran a la siguiente iteracion
superv<-extincion(n_mdls,ranking,apply(cd,1,sum))
superv<-superv[order(superv)]
#calculando estadisticas para el usuario
print(paste0("Numero de modelos evaluados: ",
             dim(poblacion)[1]-n_mdls, collapse = " "))
poblacion2<-poblacion[superv,]
objetivos2<-objetivos[superv,]
poblacion<-poblacion2
objetivos<-objetivos2
rm(poblacion2,objetivos2)
ranking<-sapply(ranking,function(x){which(superv %in% x)})
quitar<-which(sapply(ranking, function(x){length(x)})==0)
if(length(quitar)!=0){
  ranking<-ranking[-quitar]
}


print(paste0("Nuevos modelos en la primera frontera : ",
             sustituciones[Tparo2], collapse = " "))

print(paste0("Total de modelos evaluados: ",
              dim(muestra)[1] , collapse = " "))

#n_mdls-length(which(c(1:n_mdls) %in% superv))
rnkIndex<-rnkIndex[superv]
objRange <- apply(objetivos, 2, max) -   apply(objetivos, 2, min)
objRange<-as.numeric(objRange)

tryCatch(
  if(min(objetivos[,1])!=Inf){pairs(objetivos, panel=panel.smooth,
                                    cex = 1, pch = 19, bg="blue",
                                    diag.panel=panel, cex.labels = 1,
                                     font.labels=1)},error=function(x){0})

print(paste0("Tiempo de ejecucion: ",
```

```
        as.integer(100*(proc.time()[3]-tiempo))/100,
       "␣seg.␣de␣", Tparo1, "␣seg."))

    if((proc.time()[3]-tiempo)>Tparo1){ generacion <- Inf}
    #fin del ciclo
    generacion <- generacion+1
  }
  #se termina la ejecucion en paralelo
  parallel::stopCluster(cl)

  #se da formato a los resultados
  ranking<-nsga2R::fastNonDominatedSorting(objetivos)

  poblacion <- poblacion[ranking[[1]],]
  objetivos <- objetivos[ranking[[1]],]

  poblacion <- matrix(as.matrix(poblacion), nrow= dim(poblacion)[1])
  poblacion <- as.data.frame(poblacion)
  rownames(poblacion) <- 1:dim(poblacion)[1]
  colnames(poblacion) <- colnames(datos)
  colnames(poblacion)[1] <- "Intercepto"

  nombresO <- colnames(objetivos)
  objetivos <- matrix(as.matrix(objetivos), nrow= dim(objetivos)[1])
  objetivos <- as.data.frame(objetivos)
  rownames(objetivos) <- 1:dim(objetivos)[1]
  colnames(objetivos) <- nombresO

  rownames(muestra) <- 1:dim(muestra)[1]


  #se arroja la primera frontera de pareto al usuario
  #junto con sus valores objetivos
  resultados <- list()
  orden <- order(apply(poblacion, 1, sum))
  resultados$MatrizPob <- poblacion[orden,]
  resultados$MatrizValObj <- objetivos[orden,]
  rm(ranking,objetivos,poblacion)

  #la muestra de modelos evaluados
  resultados$Muestra<-muestra
  rm(muestra)
  return(resultados)
  options(warn = oldw)
}
```

valuar

```
valuar<-function(adn,datos){
  #Se inicializan los valores de la valuacion como 0
  #(valores por default)
  x1<-x2<-0

  #si adn es un objeto de la clase adn se extrae su soporte
  if (base::class(adn)=="lm"){
    modelo<-adn
    adn<-extraer_adn(adn,nombres = colnames(datos))
  }else{
    #en caso contrario se crea un modelo para poder evaluar el mismo
    if(sum(adn)<=Matrix::rankMatrix(datos[-c(1)])[1]){
      modelo<-modelar(adn,datos)
    }else{x1<-x2<-Inf}
  }

  if(x1==0){
    #evaluando el modeo
```

```
    x1<-stats::deviance(modelo)
    x2<-sum(adn)
  }

  x<-c(x1,x2)
  names(x) <- c("Devianza","NumParametros")
  return(x)
}
```

valuar_conIntercepto

```
valuar_conIntercepto<-function(adn,datos){
  #Se inicializan los valores de la valuacion como 0
  #(valores por default)
  x1<-x2<-0

  #si adn es un objeto de la clase adn se extrae su soporte
  if (base::class(adn)=="lm"){
    modelo<-adn
    adn<-extraer_adn(adn,nombres = colnames(datos))
  }else{
    #en caso contrario se crea un modelo para poder evaluar el mismo
    if(sum(adn)<=Matrix::rankMatrix(datos[-c(1)])[1]){
      modelo<-modelar(adn,datos)
    }else{x1<-x2<-Inf}
  }

  #evaluando las restricciones
  if(adn[1]==0){x1<-x2<-Inf}

  if(x1==0){
    #evaluando el modeo
    x1<-stats::deviance(modelo)
    x2<-sum(adn)
  }

  x<-c(x1,x2)
  names(x) <- c("Devianza","NumParametros")
  return(x)
}
```

valuar_sinIntercepto

```
valuar_sinIntercepto<-function(adn,datos){
  #Se inicializan los valores de la valuacion como 0
  #(valores por default)
  x1<-x2<-0

  #si adn es un objeto de la clase adn se extrae su soporte
  if (base::class(adn)=="lm"){
    modelo<-adn
    adn<-extraer_adn(adn,nombres = colnames(datos))
  }else{
    #en caso contrario se crea un modelo para poder evaluar el mismo
    if(sum(adn)<=Matrix::rankMatrix(datos[-c(1)])[1]){
      modelo<-modelar(adn,datos)
    }else{x1<-x2<-Inf}
  }

  #evaluando las restricciones
  if(adn[1]==1){x1<-x2<-Inf}

  if(x1==0){
    #evaluando el modeo
```

```
    x1<-stats::deviance(modelo)
    x2<-sum(adn)
  }

  x<-c(x1,x2)
  names(x) <- c("Devianza","NumParametros")
  return(x)
}
```

valuar_ad

```
valuar_ad<-function(adn,datos){
  #Se inicializan los valores de la valuacion como 0
  #(valores por default)
  x1<-x2<-0

  #el valor de lamda es pequenio para que las betas
  # estimadas sean cercanas al estimador por mco
  lamda <- 0.0000001
  #si adn es un objeto de la clase adn se extrae su soporte
  if (base::class(adn)=="lm"){
    modelo<-adn
    adn<-extraer_adn(adn,nombres = colnames(datos))
  }else{
    #en caso contrario se crea un modelo para poder evaluar el mismo
    if(sum(adn)<=2){
      modelo<-modelar(adn,datos)
      x1<-stats::deviance(modelo)
    }else{
        modelo<-glmnet::glmnet(
          as.matrix(datos[,-c(1,1+which(adn[-c(1)]==0))]),
          datos[,c(1)],
          lambda = lamda,
          intercept = if(adn[1]==0){FALSE}else{TRUE})
    }
    Y<-predict(modelo,newx=as.matrix(datos[,-c(1,1+which(adn[-c(1)]==0))]))
    x1 <- sum((datos[,1]-Y)^2)
  }

    #evaluando el modeo
    x2<-sum(adn)

  x<-c(x1,x2)
  names(x) <- c("Devianza","NumParametros")
  return(x)
}
```

Código de las simulaciones

```
#' @title Simulaciones
#' @description A continuacion se muestra el codigo que genera simulaciones
#' del RNSGAII para compararlo con algunos metodos y criterios de seleccion
#'  de modelos en analisis de regresion.
#' @examples
#Se instalan los paquetes necesarios
RNSGA2::instalar_paquetes()
 #analisis es un data frame que contiene las estadisticas de cada ejecucion.
analisis<-as.data.frame(matrix(nrow=1,ncol=68))
colnames(analisis)<-c("Semilla", "Metodo","Tiempo","Intercepto","MD_Ortonormal",
"RMM","M","SCRes","M_AIC","AIC","M_BIC","BIC","M_CP","CP","M_AMEMIYA","AMEMIYA",
paste0("M",0:51))
#semilla que fija las simulaciones
semilla <-1
#numero de variables
```

```
n_var <-50
#numero de observaciones
tm <- 100
#indice que corre sobre las filas de la matriz de analisis
k<-1
#se creara una funcion para crear una matrix con el numero de variables
#y cuatro criterios de seleccion de modelos: AIC, BIC, CP, el criterio
#de Amemiya y la suma de cuadrados de los residuales.
criterios<-function(modelos){
  NumVar <- sapply(modelos,function(modelo){
                sum(RNSGA2::extraer_adn(modelo,colnames(datos)))})

   AIC <- sapply(modelos, olsrr::ols_aic)

   BIC <- sapply(modelos, function(x){olsrr::ols_sbic(x,
                RNSGA2::modelar(rep(1,n_var+1),datos))})

   CP <- sapply(modelos, function(x){olsrr::ols_mallows_cp(x,
                RNSGA2::modelar(rep(1,n_var+1),datos))})

   AMEMIYA <- sapply(modelos, olsrr::ols_apc)

   SCE <- sapply(modelos, function(modelo){sum(modelo$residuals^2)})

  return(cbind(NumVar,AIC,BIC,CP,AMEMIYA,SCE))
}
#iniciando la ejecucion
while(semilla <= 100){
  ##########################################################################
  #                       SELECCION DE LA SEMILLA                          #
  ##########################################################################
  set.seed(semilla)
  ##########################################################################
  #                   SIMULACION DEL MODELO DE REGRESION                   #
  ##########################################################################

  if((semilla %% 4)==1){

    # 1: SE SIMULA UN MODELO SIN INTERCEPTO CON MATRIZ DE DISENO ORTOGONAL
    analisis$Semilla[k] <- semilla
    analisis$Intercepto[k] <- FALSE
    analisis$MD_Ortonormal[k] <- TRUE

    #se simulan los parametros del modelo
    adn <- c(0,rbinom(n_var,1,runif(1)))
    betas <- rnorm(n_var+1)*adn

    #y generan los datos del mismo sin correlaciones entre covariables
    datos <- RNSGA2::generar_datos(betas,adn,tm,Sigma = diag(n_var))
    datos <-as.data.frame(datos)
    datos$y <- datos$y - mean(datos$y)
  }

  if((semilla %% 4)==2){
    # 2: SE SIMULA UN MODELO CON INTERCEPTO CON MATRIZ DE DISENO ORTOGONAL

    analisis$Semilla[k] <- semilla
    analisis$Intercepto[k] <- TRUE
    analisis$MD_Ortonormal[k] <- TRUE

    #se simulan los parametros del modelo
    adn <- c(1,rbinom(n_var,1,runif(1)))
    betas <- rnorm(n_var+1)*adn

    #y generan los datos del mismo sin correlaciones entre covariables
    datos <- RNSGA2::generar_datos(betas,adn,tm,Sigma = diag(n_var))
    datos <-as.data.frame(datos)
    #datos$y <- datos$y - mean(datos$y)
```

```
}

if((semilla %% 4)==3){
  # 3: SE SIMULA UN MODELO SIN INTERCEPTO CON MATRIZ DE DISENO GENERAL

  analisis$Semilla[k] <- semilla
  analisis$Intercepto[k] <- FALSE
  analisis$MD_Ortonormal[k] <- FALSE

  #se simulan los parametros del modelo
  adn <- c(0,rbinom(n_var,1,runif(1)))
  betas <- rnorm(n_var+1)*adn

  #y generan los datos del mismo con correlaciones entre covariables
  A<-matrix(rcauchy(n_var^2), n_var)
  Sigma <- t(A)%*%A
  Sigma<-sapply(1:dim(Sigma)[1], function(i){
    sapply(1:dim(Sigma)[1], function(j){
      Sigma[i,j]/(sqrt(Sigma[i,i]*Sigma[j,j]))
    })})

  datos <- RNSGA2::generar_datos(betas,adn,tm,Sigma = Sigma)
  datos <-as.data.frame(datos)
  datos$y <- datos$y - mean(datos$y)
}

if((semilla %% 4)==0){
  # 4: SE SIMULA UN MODELO CON INTERCEPTO CON MATRIZ DE DISENO GENERAL

  analisis$Semilla[k] <- semilla
  analisis$Intercepto[k] <- TRUE
  analisis$MD_Ortonormal[k] <- FALSE

  #se simulan los parametros del modelo
  adn <- c(1,rbinom(n_var,1,runif(1)))
  betas <- rnorm(n_var+1)*adn

  #y generan los datos del mismo con correlaciones entre covariables
  A<-matrix(rcauchy(n_var^2), n_var)
  Sigma <- t(A)%*%A
  Sigma<-sapply(1:dim(Sigma)[1], function(i){
    sapply(1:dim(Sigma)[1], function(j){
      Sigma[i,j]/(sqrt(Sigma[i,i]*Sigma[j,j]))
    })})

  datos <- RNSGA2::generar_datos(betas,adn,tm,Sigma = Sigma)
  datos <-as.data.frame(datos)
  #datos$y <- datos$y - mean(datos$y)
}

#estimamos el modelo simulado
modelo <- RNSGA2::modelar(adn, datos)
analisis$M[k] <- sum(adn)
analisis$SCRes[k] <- sum(modelo$residuals^2)
rm(modelo)

#Estimacion Backward
analisis$Metodo[k]<-"Backward"
tiempos<-system.time(
  backward <- leaps::regsubsets(y~., data = datos,
                                method="backward",
                                nvmax=n_var,
                                intercept=analisis$Intercepto[k],
                                really.big=TRUE)
)
backward <- summary(backward)$which

#evaluando el tiempo de ejecucion
```

```
  analisis$Tiempo[k] <- tiempos[1]

  #calculando el rango de la matriz de modelos
  analisis$RMM[k] <- Matrix::rankMatrix(backward)

  #calculando los modelos
  modelos_backward <- apply(backward,1,
                            function(x){RNSGA2::modelar(
                              if(analisis$Intercepto[k]){x}else{c(0,x)},
                              datos)})

  #se evaluan los criterios de seleccion
  MatrizCriterios <- criterios(modelos_backward)
  MatrizCriterios <- as.data.frame(MatrizCriterios)

  #se agregan las sumas de cuadrados de los residuales a la tabla de analisis
  analisis[k,
           sapply(paste0("M",MatrizCriterios$NumVar),
           function(nombre){which(nombre==colnames(analisis))})] <- MatrizCriterios$SCE

  #se agregan los valores del modelo seleccionado
  analisis[k,
           sapply(colnames(MatrizCriterios[,-c(1,6)]),
                  function(nombre){
                    which(colnames(analisis)==nombre)})
           ] <- apply(MatrizCriterios[,-c(1,6)], 2, min)

  #y su numero de variables
  analisis[k,
           sapply(c("M_AIC","M_BIC","M_CP","M_AMEMIYA"),
                  function(nombre){
                    which(colnames(analisis)==nombre)})
           ] <- apply(MatrizCriterios[,-c(1,6)],
                      2,
                      function(valor){
                        MatrizCriterios$NumVar[which.min(valor)]
                      })

  k<-k+1
  analisis[k,c(1,4,5,7,8)]<-analisis[k-1,c(1,4,5,7,8)]

  #Estimacion por busqueda exhaustiva
  analisis$Metodo[k]<-"Exhaustivo"
  tiempos<-system.time(
    regsubset <- leaps::regsubsets(y~.,
                                   data = datos,
                                   method="exhaustive",
                                   nvmax=n_var,
                                   intercept=analisis$Intercepto[k],
                                   really.big=TRUE)
  )
  regsubset <- summary(regsubset)$which

  #calculando los modelos
  modelos_regsubset <- apply(regsubset,1,
                             function(x){RNSGA2::modelar(
                               if(analisis$Intercepto[k]){x}else{c(0,x)},
                               datos)})

  #evaluando el tiempo de ejecucion
  analisis$Tiempo[k] <- tiempos[1]

#se evaluan los criterios de seleccion
  MatrizCriterios <- criterios(modelos_regsubset)
  MatrizCriterios <- as.data.frame(MatrizCriterios)

  #se agregan las sumas de cuadrados de los residuales a la tabla de analisis
  analisis[k,
```

```
          sapply(paste0("M",MatrizCriterios$NumVar),
          function(nombre){
                      which(nombre==colnames(analisis))})] <- MatrizCriterios$SCE

#se agregan los valores del modelo seleccionado
analisis[k,
          sapply(colnames(MatrizCriterios[,-c(1,6)]),
                 function(nombre){
                   which(colnames(analisis)==nombre)})
          ] <- apply(MatrizCriterios[,-c(1,6)], 2, min)

#y su numero de variables
analisis[k,
          sapply(c("M_AIC","M_BIC","M_CP","M_AMEMIYA"),
                 function(nombre){
                   which(colnames(analisis)==nombre)})
          ] <- apply(MatrizCriterios[,-c(1,6)],
                     2,
                     function(valor){
                       MatrizCriterios$NumVar[which.min(valor)]
                     })

k<-k+1
analisis[k,c(1,4,5,7,8)]<-analisis[k-1,c(1,4,5,7,8)]

#Estimacion Forward
analisis$Metodo[k]<-"Forward"
tiempos<-system.time(
  forward <- leaps::regsubsets(y~., data = datos,
                               method="forward",
                               nvmax=n_var,
                               intercept=analisis$Intercepto[k],
                               really.big=TRUE)
)
forward <- summary(forward)$which

#calculando los modelos
modelos_forward <- apply(forward,1,
                         function(x){RNSGA2::modelar(
                           if(analisis$Intercepto[k]){x}else{c(0,x)},
                           datos)})

#evaluando el tiempo de ejecucion
analisis$Tiempo[k] <- tiempos[1]

#se evaluan los criterios de seleccion
MatrizCriterios <- criterios(modelos_forward)
MatrizCriterios <- as.data.frame(MatrizCriterios)

#se agregan las sumas de cuadrados de los residuales a la tabla de analisis
analisis[k,
          sapply(paste0("M",MatrizCriterios$NumVar),
              function(nombre){
                      which(nombre==colnames(analisis))})] <- MatrizCriterios$SCE

#se agregan los valores del modelo seleccionado
analisis[k,
          sapply(colnames(MatrizCriterios[,-c(1,6)]),
                 function(nombre){
                   which(colnames(analisis)==nombre)})
          ] <- apply(MatrizCriterios[,-c(1,6)], 2, min)

#y su numero de variables
analisis[k,
          sapply(c("M_AIC","M_BIC","M_CP","M_AMEMIYA"),
                 function(nombre){
                   which(colnames(analisis)==nombre)})
          ] <- apply(MatrizCriterios[,-c(1,6)],
```

```
                    2,
                    function(valor){
                      MatrizCriterios$NumVar[which.min(valor)]
                    })

k<-k+1
analisis[k,c(1,4,5,7,8)]<-analisis[k-1,c(1,4,5,7,8)]

#Estimacion por NSGA-II
analisis$Metodo[k]<-"NSGA-II"
tiempos<-system.time(
  NSGAII<-RNSGA2::RNSGAII(
    funcion.fit = if(analisis$Intercepto[k]){
      "valuar_conIntercepto"}else{
        "valuar_sinIntercepto"
      },
    datos=datos,
    cl = parallel::makeCluster(parallel::detectCores())
  ))

#calculando los modelos
modelos_nsgaii <- apply(NSGAII$MatrizPob,
                        1,
                        function(x){
                          RNSGA2::modelar(x,datos)})

#evaluando el tiempo de ejecucion
analisis$Tiempo[k] <- tiempos[1]

#se evaluan los criterios de seleccion
MatrizCriterios <- criterios(modelos_nsgaii)
MatrizCriterios <- as.data.frame(MatrizCriterios)

#se agregan las sumas de cuadrados de los residuales a la tabla de analisis
analisis[k,
         sapply(paste0("M",MatrizCriterios$NumVar),
                function(nombre){
                  which(nombre==colnames(analisis))})] <- MatrizCriterios$SCE

#se agregan los valores del modelo seleccionado
analisis[k,
         sapply(colnames(MatrizCriterios[,-c(1,6)]),
                function(nombre){
                  which(colnames(analisis)==nombre)})
         ] <- apply(MatrizCriterios[,-c(1,6)], 2, min)

#y su numero de variables
analisis[k,
         sapply(c("M_AIC","M_BIC","M_CP","M_AMEMIYA"),
                function(nombre){
                  which(colnames(analisis)==nombre)})
         ] <- apply(MatrizCriterios[,-c(1,6)],
                    2,
                    function(valor){
                      MatrizCriterios$NumVar[which.min(valor)]
                    })

k<-k+1
analisis[k,c(1,4,5,7,8)]<-analisis[k-1,c(1,4,5,7,8)]

#Estimacion Reemplazo Secuencial
analisis$Metodo[k]<-"Reemplazo␣Secuencial"
tiempos<-system.time(
  seqrep <- leaps::regsubsets(y~., data = datos,
                              method="seqrep",
                              nvmax=n_var,
                              intercept=analisis$Intercepto[k],
                              really.big=TRUE)
```

```
  )

  seqrep <- summary(seqrep)$which

  #calculando los modelos
  modelos_seqrep <- apply(seqrep,1,
                          function(x){RNSGA2::modelar(
                            if(analisis$Intercepto[k]){x}else{c(0,x)},
                            datos)})

  #se evaluan los criterios de seleccion
  MatrizCriterios <- criterios(modelos_seqrep)
  MatrizCriterios <- as.data.frame(MatrizCriterios)

  #evaluando el tiempo de ejecucion
  analisis$Tiempo[k] <- tiempos[1]

  #se agregan las sumas de cuadrados de los residuales a la tabla de analisis
  analisis[k,
           sapply(paste0("M",MatrizCriterios$NumVar),
                  function(nombre){
                    which(nombre==colnames(analisis))})] <- MatrizCriterios$SCE

  #se agregan los valores del modelo seleccionado
  analisis[k,
           sapply(colnames(MatrizCriterios[,-c(1,6)]),
                  function(nombre){
                    which(colnames(analisis)==nombre)})
           ] <- apply(MatrizCriterios[,-c(1,6)], 2, min)

  #y su numero de variables
  analisis[k,
           sapply(c("M_AIC","M_BIC","M_CP","M_AMEMIYA"),
                  function(nombre){
                    which(colnames(analisis)==nombre)})
           ] <- apply(MatrizCriterios[,-c(1,6)],
                      2,
                      function(valor){
                        MatrizCriterios$NumVar[which.min(valor)]
                      })

  #salvamos los resultados
  write.csv(analisis,file="~/analisis.csv")
  k<-k+1
  analisis[k,] <- rep(NA,dim(analisis)[2])

  if(semilla<10){
    save.image(paste0("~/Simulacion_00",semilla,".RData"))
  }else{
    if(semilla<100){
      save.image(paste0("~/Simulacion_0",semilla,".RData"))
    }else{
      save.image(paste0("~/Simulacion_",semilla,".RData"))
    }
  }

  semilla<-semilla+1
}
```

Bibliografía

[1] Barron, A. R. (1989). *Statistical properties of artificial neural networks.* Proceedings of the IEEE International Conference on Decision and Control. P.p. 280–285.

[2] Box, G. E., Jenkins, G. M., Reinsel, G. C., Ljung, G. M. (2015). *Time series analysis: forecasting and control.* EUA. John Wiley & Sons.

[3] Breusch, T. S., Pagan, A.R. (1979). *A simple test for heteroscedasticity and random coefficient variation.* Econometrica: Journal of the Econometric Society. P.p 1287-1294.

[4] Chaman, I. C. (2012). *Uso de un Algoritmo de Cúmulo de Partículas Basado en Hipervolumen para Resolver Problemas Multi-Objetivo.* Ciudad de México: Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional.

[5] Durbin, J., Watson, G.S. (1950). *Testing for serial correlation in least squares regression I.* Biometrika. Vol. 37, no. 3/4, p.p 409-428.

[6] Foster, D. P., George, E. I. (1994). *The risk inflation criterion for multiple regression.* The Annals of Statistics. P.p. 1947–1975.

[7] Friedman, J., Hastie, T., Tibshirani, R. (2001). *The elements of statistical learning.* Vol. 7, no.10. New York: Springer series in statistics.

[8] Golchha, A., Qureshi, S. G. (2015). *Non-Dominated Sorting Genetic Algorithm-II – A Succinct Survey.* International Journal of Computer Science and Information Technologies. Vol. 6, no. 1, p.p. 252-255.

[9] Herlands, W., De-Arteaga, M., Neill, D., Dubrawski, A. (2015). *Lass-0: sparse non-convex regression by local search.* https://arxiv.org/abs/1511.04402.

[10] Icke, I., Bongard, J. C. (2013). *Improving genetic programming based symbolic regression using deterministic machine learning.* IEEE Congress on Evolutionary Computation. Pp. 1763-1770. IEEE.

[11] James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). *An introduction to statistical learning*. Vol. 6. New York: springer.

[12] Lambrinos, J. (1984). *Applied Linear Regression Models*. Taylor & Francis.

[13] Määttä, J. (2016). *Model Selection Methods for Linear Regression and Phylogenetic Reconstruction*.

[14] Marjanovic, G., Ulfarsson, M. O., Hero, A. O. (2015). *MIST: l_0 Sparse Linear Regression with Momentum*. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). P.p. 3551-3555. IEEE.

[15] McCullagh, P., Nelder, J. A. (1989). *Generalized linear models*. Monographs on Statistics and Applied Probability. Springer US.

[16] Paterlini, S., Minerva, T. (2010). *Regression model selection using genetic algorithms*. Proceedings of the 11th WSEAS international conference on nural networks and 11th WSEAS international conference on evolutionary computing and 11th WSEAS international conference on Fuzzy systems. P.p. 19–27. World Scientific and Engineering Academy and Society (WSEAS).

[17] Sharpiro, S. S., Wilk, M.B. (1965). *An analysis of variance test for normality*. Biometrika. Vol. 52, no. 3/4, p.p. 591-611.

[18] Shor, N. Z. (1985). *Minimization Methods for Non-differentiable Functions*. Springer Series in Computational Mathematics. Vol. 3. Springer Science & Business Media.

[19] Soler, M. (2002). *Evolución*. Proyecto Sur de Ediciones: Granada.

[20] Tibshirani, R. J., (2013). *The lasso problem and uniqueness*. Electronic Journal of statistics. Vol. 7, p.p. 1456-1490.

[21] White, H. (1980). *A heteroskedasticity-consistent covariance matrix estimator and direct test for heterocedasticity*. Econometrica: journal of the Econometric Society. P.p 817-838.